AN ABSTRACT OF THE THESIS OF

Carrie Rebhuhn for the degree of Master of Science in Mechanical Engineering

presented on May 13, 2013.

Title: A Multiagent Approach to Identifying Innovation in Design Components

Abstract approved: _____

Kagan Tumer

Innovation is a key element for a product to achieve market success, but identifying it within product or even defining the term is a difficult task. Identifying innovation has been approached in many different ways. Experts in design engineering may identify innovative designs based on an analysis of a product's functions, and statistical techniques may be used to evaluate innovation within a set of products. While there are numerous ways to recognize innovation in a *product*, there is no straightforward way of identifying how much each *component* within a product contributes to its innovation.

Multiagent systems face an analogous problem; though the performance of a system may be easily assessed, the complex interactions of the agents makes using this system performance to reward each agent ineffective. Difference rewards provide a mechanism for a multiagent system to better quantify the impact of an agent on the system's performance.

We introduce the Creative Agents for Repository-Reliant Innovation Engineering (CARRIE) algorithm, which frames the problem of creating a design as supervised learning within a multiagent system. Agents simulate the design process by selecting components to create a product from their training data, and receive external evaluations based on the product-level innovation score. In order to propagate this score to the component selections, the CARRIE algorithm incorporates difference rewards to identify components that positively or negatively impact the overall innovation score within a set of products.

Traditional application of the difference reward requires a way to calculate a system's performance, and then a way to recalculate this performance when an agent is removed in simulation. This presents a problem when we only have the numerical evaluation of the innovation in a product to use as a system performance score, and no indication of how this innovation score was obtained. For this reason, the CARRIE algorithm uses a method by which we can calculate the system score based on the novelty scores of the components in a product. This enables the computation of the difference reward in this domain without actually having a mathematical formulation of an arbitrary system reward.

A Multiagent Approach to Identifying Innovation in Design Components

by

Carrie Rebhuhn

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented May 13, 2013
Commencement June 2013

Master of Science thesis of Carrie Rebhuhn presented on May 13, 2013.

APPROVED:

Major Professor, representing Mechanical Engineering

Director of the School of Mechanical, Industrial, and Manufacturing Engineering

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Carrie Rebhuhn, Author

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Kagan Tumer, for his pep talks, guidance, and all of the brainpower he has given toward helping me achieve my goals. I would also like to thank Dr. Irem Tumer for her continuing support and for starting me on this path. Without her influence this thesis would not exist.

I would like to also acknowledge my collaborators on this project, Dr. Sarah Oman and Brady Gilchrist, for laying the groundwork for this thesis and helping me to better understand the concepts of creativity and innovation. I would also like to acknowledge Dr. Rob Stone for his expertise with the Design Repository and thoughtful discussions on interesting research directions for this project.

I also wish thank Dr. Matt Knudson, for his hours in the lab dedicated to instructing, philosophizing, and debating with me. He first caught my interest with multiagent systems and coding, and has proved to be an excellent mentor and loyal friend.

I would particularly like to thank (the future Dr.) William Curran, first for proof-reading this thesis, but mostly for giving me enough confidence to send the first draft off to Kagan. I am also grateful to my other friends in the AADI lab for the banter, the discussion, the happy hours, the all-nighters, the coffee runs, the whiteboard debates, and all of the other ways in which they have shaped my graduate school experience.

Finally, I would like to thank my family for their all of their encouragement and support throughout the years.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

Figure Page

LIST OF FIGURES (Continued)

Figure Page

LIST OF FIGURES (Continued)

Figure Page

LIST OF TABLES

LIST OF APPENDIX FIGURES

Figure Page

LIST OF APPENDIX TABLES

Chapter 1: Introduction

Innovation in product design can revolutionize the way that we use and think about products, and can increase the marketability of new products. Companies that develop new creative ways to design products have the chance to corner the market on their concept. But the development of innovative designs is commonly based on the designer's experience or the experience of their company. For engineers to acquire this experience through conventional means, such as research or training, takes up time and money, and provides the engineer with a limited scope of knowledge to draw upon. When competing for patents on new and innovative designs, the process of researching new methods of design must be streamlined to be successful. This acquisition of knowledge about innovative new techniques may be enhanced by incorporating computer-aided design techniques. By focusing on innovation early in the design process, the potential success of different concepts may be evaluated and the finalized design can be executed with fewer revisions and lower prototyping cost.

Automatic concept generation has proved to be a promising avenue of design research. Previously concept generation has focused on generating component suggestions which are most likely to work based on their functional role in historical design data [3]. While this approach has proven helpful in generating a diverse set of designs, it does not specifically target the promotion of innovation in the resultant design. The technique of automatic concept generation has room for expansion; if a way to score the innovation of particular *components* may be identified, then the suggestions given by the generator may be sorted in order of most innovative to least innovative.

Innovation is difficult to evaluate. An analysis of patents may show the innovativeness of companies [1], and market success may play a part in identifying innovation in products, but currently there is no way to objectively calculate the innovation of a product from the product's construction alone. Even an evaluation of a product's innovation will vary from person to person based on their experience. Nonetheless we can still identify innovation subjectively; consumer magazines publish yearly lists of products which they identify as innovative. As with any human evaluation, bias must be considered, but

with sufficient data a consensus may be reached on the innovation of a product.

While the identification of innovative products is possible, the basis for *scoring* that innovation is still an open topic. Consumer reports may identify a design which is innovative, but there is no standard metric for innovation yet in use. Additionally, in order to promote innovation during the design stage, this scoring mechanism needs to not only apply to fully-formed products, but somehow also to their components or structure. Work by Oman et al. [17] has provided a method of using the frequency that a function appears in a design set to estimate the creativity of a full product using a weighted sum of the novelty scores of those functions. Additionally, work by Oman et al. also focused on using a functional subtraction method with product pairs to identify what made some designs stand out from the standard version. However, no objectively-calculable link has yet been made between the innovation score of an entire product and the innovation score at the component level.

Multiagent reinforcement learning systems face the analogous problem of identifying how each member contributes to the performance of the system. In cooperative multiagent reinforcement learning, each agent takes an action and receives a reward for this action. Determining the reward that an agent receives is not always a straightforward process. It seems intuitive that each agent should be given the system reward to promote cooperation, but this does not always produce the desired behavior. When an agent takes an action with many other agents, the total reward of the entire system may not reflect the contribution of the single agent. For example, if many agents improved the system performance, but the single agent decreased it only slightly, the poorly-performing agent would still receive a positive reward. In order to design rewards which promote desirable learning behavior, these rewards may be adjusted through a process called *reward shaping*. The reward shaping technique known as *difference rewards* have been used as a simple mechanism to combat this confusing reward signal by calculating only the portion of the system reward to which the agent directly contributed. The subtraction method used by Oman et al. is similar to the concept of difference rewards, although it is performed at a functional level rather than a component level.

A design may be framed as a complex multiagent system, with many agents (representing the functions performed by the design) working together to contribute to the innovation score of the design. The correct reward for each agent in this multiagent system is particularly difficult to determine, because we cannot directly calculate how the

presence of some components or functions impacts the score at the product level. The application of difference rewards has been successful in a diverse set of complex systems such as network routing [22] and rover coordination [9]. Network routing is somewhat similar to this problem; a system of interconnected nodes perform routing operations and then receive a performance measure based on the total throughput of the network. Like the network routing domain, the design of a product consists of interconnected functions, which are performed by components within the design, and the design receives a performance measure based on its innovation.

We may not have a method of calculating the innovation of a product, but we can assess it through human methods such as surveys or expert identification. This human evaluation process falls short in that it does not give a mathematical forward calculation for the product-level reward. Difference rewards rely on the calculation of the system-level reward minus the system-level reward if the agent had not participated. Typically a recalculation of the system-level reward is necessary to obtain the system reward without the agent. Approximate methods may be admissible, however, as it has been shown in the air traffic domain [18] that an approximation of the calculation of a system reward is often a sufficient and preferential to performing this recalculation.

We introduce the Creative Agents for Repository-Reliant Innovation Engineering (CARRIE) algorithm, which approximates the effect of removing the agent by decomposing the system score by novelties, and uses this in as a difference calculation. In this way we can use a multiagent reinforcement learning framework with the difference reward to propagate the product-level evaluation of the innovation scores down to the component level. By structuring the design process as a supervised learning task, the CARRIE algorithm learns innovation-based values for selecting each of the components found in a design. By using the functional information present in the Design Repository and external scorings of the products together as a training set, the CARRIE algorithm learns the innovative impact of different components.

Our approach is facilitated by the wealth of data available from the Design Repository, housed at Oregon State University, which represents a collection of products which have been cataloged and decomposed into their functional elements. A major feature of the design repository is its ability to provide suggestions of components for use in future designs based on their appearance in past products, making it an ideal tool for offering design suggestions based on innovation scores.

The CARRIE algorithm takes in an external product-level innovation score and propagates this score down to the component level. We show that our method of component-level evaluation fits better with the product-level evaluation than novelty, a previous metric used to identify innovation. We then demonstrate how these innovation scores can be used to improve the design of future products.

1.1 Terminology

The foundation of the techniques used here lie in both design engineering and multiagent systems. There are a few words have specific meanings within this work, which are explained in this section. In particular the words *novelty*, *creativity*, and *innovation* are commonly used to describe a similar quality, but have disparate meanings in the design literature. Here we take *novelty* as the mathematical description of the frequency of a component in the dataset, as given by Shah et al. in [19]. *Creativity* as we define it refers to the ability for a design methodology to generate new and unconventional design ideas, while *innovation* refers to the actualization of the creative design suggestions in a final product.

We also use several terms which have specific meanings within the context of this work but whose definitions may vary in other work. *Products* are designs from the repository from which a function-component matrix is gathered. Information on the products (such as pictures, descriptions, etc) is also used to obtain an external evaluation of the innovation of the whole product. *Functions* are capabilities of the products. *Components* are the physical parts of the product which are cataloged in the design repository. The number of times a component performs a function within the design is also recorded, but this information is not used in this approach.

We also draw upon concepts from the field of multiagent learning. *Agents* are learning elements of a multiagent system. Each agent keeps an internal evaluation of the actions available to it, and makes decisions based on these evaluations. *Rewards* are a numerical evaluation given to the agent for a particular action it takes, and using these rewards the agent can adjust its internal evaluation of the action that it has taken. Further concepts related to this will be explained in Section 2.1.

1.2 Contributions of This Thesis

The contributions of this thesis are primarily to lay the groundwork for component-level evaluation of innovation in products. We cannot verify our results or prove that they are better than the golden standard here because no golden standard yet exists for innovation propagation. We do show that these results may be used to identify components that have, in the design set analyzed, contributed to the innovativeness of the products in which they are present. There are four main contributions of this thesis:

- In Section 3, we introduce the Creative Agents for Repository-Reliant Innovation Engineering (CARRIE) algorithm, a multiagent-based algorithm with which we can learn innovation scorings at the component level. This frames the process of design engineering as a multiagent system, and assigns agents to fulfill functional requirements of the design.

- In Section 4, we provide the difference reward used by the CARRIE algorithm with two variations in suggested counterfactual. This is a unique application of the difference reward because the counterfactual term is impossible to calculate. We assume an underlying structure to the product-level innovation score which relates to the weighted novelty summation of each component.

- In Section 5, we present results from analysis on three different datasets using the CARRIE algorithm with our two difference rewards. We show trends in component-level innovation which have previously not been identified.

- In Section 6, we show a novel application of this work toward the innovation-targeted redesign of an existing product. This helps to highlight the potential use of the data gathered using the CARRIE algorithm, and how it might be used with an automatic concept generator to rank design suggestions based on their estimated innovativeness.

Chapter 2: Background

In this section we provide background information which is key to the understanding the development and execution of the CARRIE algorithm. Because this work is a hybrid of design engineering and artificial intelligence, the background covers the tools we use from both fields. The reader who is interested in multiagent systems and is already familiar with the functionality of the Design Repository may skip to Section 2.1. The reader who has an interest in learning more about the design techniques tools and influences in this paper, and who perhaps has some background in work in multiagent reinforcement learning, may refer to Section 2.2.1. Section 2.3 is meant to provide background on the scoring of product innovation using statistical techniques, but it is not covered in depth in this work because it represents only a potential input to the CARRIE algorithm rather than a necessary part of the algorithm's structure.

2.1 Multiagent Systems

Multiagent reinforcement learning is a field of artificial intelligence that involves the coordination of distributed learners. Control of a complex system may be modeled as a set of less complex interacting entities which make autonomous decisions. These entities are called 'agents', and under a reinforcement learning framework they interact with the world through a process of sensing the world state, taking actions available within that state, and receiving rewards. Agents may have other methods of adaptation, including neural networks and evolutionary algorithms, but we focus in this work on reinforcement learning because of its simple structure and similarity to the organic adaptation process.

Reinforcement learning in a cooperative multiagent system focuses on finding a set of actions which most benefits the collective system. Learning agents adapt their strategies through repeatedly taking actions and getting a reward for these actions. The Design Repository provides a large database over which to train agents in a supervised learning problem, which will be explained further in 2.1.2. The mechanism for reinforcement learning is covered in the next section.

2.1.1 Action-Value Update

An agent learns a *policy*, which holds information on an agent's perception of the value of taking a particular action. This policy is updated each time the agent receives a reward for an action it has taken. We perform *action-value update*. An agent is therefore rewarded for taking an action and receiving a reward. This value is updated with each new observation from the agent, according to the equation:

$$V(a) \leftarrow V(a)_{old} + \alpha(R(a) - V(a)_{old}) \tag{2.1}$$

where $V(a)$ is the expected value of the action, α is the learning rate, and $R(a)$ is the reward received for taking action a. α is a number on the range [0,1] which impacts the learning speed of the agent. Increasing the α parameter may increase the value that an agent puts on more recent information, while decreasing α lowers learning speed but increases the chance of long-term convergence to an optimal policy.

In a typical reinforcement learning system, agents use their knowledge about the value of specific actions to select their next action, via either ϵ-greedy or softmax action selection methods. Due to the fact that we cannot evaluate an arbitrary design we constrain our exploration in a different way; we force action selection to match the products we have and then reward accordingly. We can evaluate the actions that the agents are forced to select using *supervised learning*.

2.1.2 Supervised Learning

Reinforcement learning may occur with varying degrees of intervention from an expert. Some reinforcement learning requires little intervention; an agent may be given a general idea of what the user wants and from this be able to discern its reward without being given an explicit value. Other reinforcement learning systems are fully supervised; a value is given directly for each action taken by the system according to its contribution toward the system goal.

Demonstration may also enhance learning within a system. Demonstration is used in robotics to get a solution close to that which is desired without explicitly hard-coding a solution for a robot [9]. In this way the robot can still be adaptable and progress toward the best solution, but won't take as long to find the space of acceptable solutions.

We use a combination of supervised learning with demonstration, which constricts the actions of the agents to be within the set of scores that we have available. In this way we can train on the datasets for which we have information. Generalization of this reward signal occurs during the learning process, as agents see several differently-scored examples of a solution and internally change their estimation of its value.

2.1.3 Reward Shaping

When a system operates under a reinforcement learning structure, agents are given rewards which relate to their goal. Sometimes it is useful to give small extra rewards in order to shape the desired system behavior. Reward shaping is the process through which a reward is modified to promote a particular behavior in the system. Complex systems are particularly benefited by reward shaping, as adding extra rewards based on prior system knowledge may offer much better performance without the added cost of an exact error computation in an agent's behavior. A successful shaping reward will preserve the tendency toward optimal performance while speeding convergence time.

Reward shaping is often designed to address the 'credit assignment problem' in multiagent learning, which suggests that a simple system-level evaluation of a reward may be insufficient to accurately communicate to the agent to what extent it influenced the system with its actions. Directly calculating the impact of an agent's action in a complex multiagent system is typically intractable because it would require tracking of the effects of the agent's decision across the entire system.

Difference rewards solve this problem in that they preserve the optimality of the converged reward, but they encapsulate *only* the agent's impact on the system. They 'shape' the reward through subtraction of irrelevant reward information. These will be discussed further in the following section.

2.1.4 Difference Rewards

A conceptually simple method of identifying an agent's effect on the system is through a difference reward. Difference rewards compare the original system reward to a calculated system reward in which an agent does not exist, or is replaced by some counterfactual, which refers to an alternative solution that the agent might have taken. Formally, this

is expressed as:

$$D_i(z) = G(z) - G(z_{-i} + c_i) \tag{2.2}$$

where $G(z)$ is the reward given to the entire system and $G(z_{-i} + c_i)$ is the reward given to a hypothetical system in which the effects of agent i were replaced by some counterfactual c_i. This addition of the counterfactual offsets the effect of the agent removal; the z_{-i} term is a system state vector with the ith element set to zero, while the counterfactual c_i is a zero-padded vector containing only a value for the ith element. When talking about a counterfactual in terms of a product design, it is similar to *replacing* a component in the design with a different component. The difference reward would then be calculated by comparing the original innovation score to the innovation score with a substituted component. The selection of this counterfactual has been explored to some degree in previous work [2], and can promote varying behavior in the system.

In this work, difference rewards are used to determine the impact of a component on the product-level innovation score. The advantages in noise reduction and learning speed are particularly desired due to the presence of a large number of agents within a system and the relatively low number of times component solutions are seen within the repository.

Previous publications using the design repository have used difference rewards as a basis for functional subtraction, which has allowed for a simple identification of functionality in products [17]. Additionally, difference rewards have been applied to a variety of multiagent systems, including air traffic control [18], network routing [22], and robot coordination [9] to promote cooperative behavior between agents.

Sometimes it is not computationally tractable to solve for a difference reward, or the system reward formulation may be unknown. This does not necessarily preclude the application of a difference reward. In the air traffic domain the difference reward has been estimated without significant detriment to the performance in order to speed simulation time [18]. Additionally, an extension of the difference reward has been developed using statistical techniques for black box problems which have no way to recalculate the system reward [21]. We face a similar problem as in this work: we have no way to recalculate the system without the influence of an agent. However, our sample size is too small to estimate the system reward reliably using a table of averages as used in [21]. Because

we have such a small sample size, we rely on a decomposition based on a component's novelty, which has been correlated to the innovation of a device in the past.

2.2 Design Engineering

The field of design engineering provides us with a unique domain for the application of multiagent techniques. In order to use these techniques as a research tool, we must have a database of knowledge from which to work. In particular, in Sections 2.2.1 and 2.2.2 we identify tools available in the design engineering world which are store and simplify data about products. In Section 2.2.3, we also discuss the topic of automatic concept generation, which provides existing methods for computer-assisted conceptual design. Section 2.2.4 gives previous approaches in the quantification of innovation and creativity at the functional level.

2.2.1 Design Repository

The Design Repository at Oregon State University contains a wealth of information about modern products. It currently contains over 180 products with more than 6000 artifacts. The Repository was created to be a database of useful design information to provide inspiration for future design [3], as well as a method to analyze failures in designs [16]. Techniques built into the Design Repository to offer design suggestions will be discussed in more detail in Section 2.2.3.

The Design Repository catalogs information on many product parameters, such as product functions, components, assemblies, mass, dimensions, materials, manufacturing processes, and failure modes. Functional models are created for every product during the physical deconstruction process, identifying the way that materials, signals, and energy change as they flow through the product.

The interface to the Design Repository is shown in Figure 2.1. Function-component models, as well as a variety of other methods of decomposing product attributes, may be selected and output in a spreadsheet format. Other information on products is also available, but this is not directly relevant to the work presented here. The interested reader may access the Design Repository at `http://designengineeringlab.org/delabsite/repository.html`.

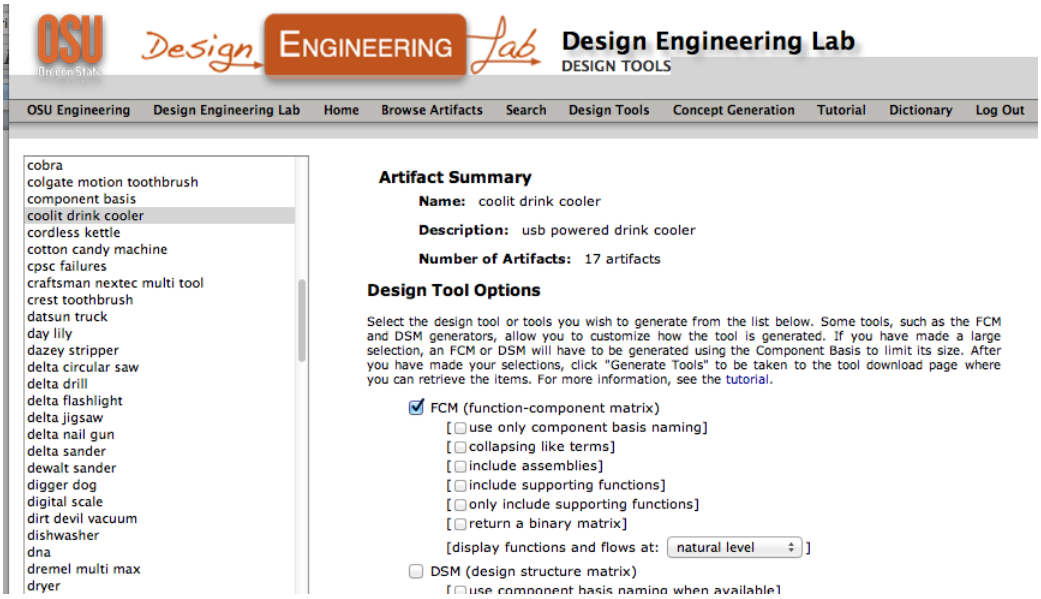

Figure 2.1: A screenshot of the Design Repository, showing the method of obtaining function-component models for this research. The component basis naming is selected to ensure function and component naming standardization.

2.2.2 Function-Component Matrices

The data contained in the Design Repository has already been instrumental in the development of concept generation, as further outlined by 2.2.3. Because there are standardized mechanisms for cataloging and outputting the data within the repository, it has the immense benefit of being readable by a simple program. This allows data handling to be automated, and data from hundreds of designs may be uploaded to a program and manipulated to analyze patterns across the repository.

The CARRIE algorithm treats the data in the Design Repository as a training set. Specifically it uses the Function-Component Matrices (FCMs) stored for each product. FCMs are used to store an abstract disassembly of the product. They map functionality found in the device to the component which served that functionality, but do not incorporate structure information. Figure 2.2 shows an example of an FCM. The columns represent components in the matrix, while the rows represent functions. A binary indicator is then used to identify connections in this matrix (0 if the component does not fill

a function, 1 if it does).

When working with a large database, the standardization of language used to describe the functions and components is essential. Functions and flows are kept consistent through the use of the functional basis [20]. The functional basis allows not only a standardization of terminology, but also offers different levels of descriptive detail to be obtained. For example, the same function may be described as "provision energy", which gives little indication of the directionality or type of energy, or as "supply electrical", which indicates that a component's utility in the system is to supply an electric current to another component. When using this as a research tool, there are benefits and drawbacks to using more or less specific language to describe functionality, as generalizability of the data can suffer with more specific language. This means that the patterns in the data will be more accurate with more specific language, but a definite pattern may not necessarily be seen if the language is too specific.

We use the most general language in this work (natural level) because the number of products we examine is not large. This allows us to have the most overlap in product functionality, and therefore we see more examples of different components performing these functions. This is important for learning, as will be discussed in Section 2.1, because reinforcement learning algorithms require many iterations seeing similar data for convergence. If the language of the data was so specific that there was no commonality between the products, the CARRIE algorithm would have only one iteration, and would have no chance to converge.

2.2.3 Automatic Concept Generators

The main interest of the Design Repository for engineers is the potential to enhance the development of new designs using data gathered by the repository. A variety of approaches have been used to leverage the information contained in the design repository, specifically to provide feasible solutions to a requirement or set of requirements. In particular, the site containing the Design Repository offers the built-in functionality of the Morphological Evaluation Machine and Interactive Conceptualizer (MEMIC).

MEMIC provides a designer with a selection of components which, based on the data in the Design Repository and the requirements given to it, provides a solution which is likely to work. MEMIC offers components which have historically been able to link

Function-Component Matrix					
Generated On:					
	battery	belt	circuit board	cover	elect
provision energy	1	0	0	0	
provision energy	1	0	0	0	
channel energy	0	1	0	0	
signal signal	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
signal energy	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
channel energy	0	0	1	0	
channel signal	0	0	1	0	
convert energy	0	0	1	0	

Figure 2.2: Example section of an FCM from the Design Repository for the Neato Robotix vacuum cleaner with data displayed at the natural level.

certain functions together. By offering the user only components which have been shown to link certain functionalities, MEMIC attempts to provide the designer with an option that has historically appeared to play the same role in other designs, but does not offer further information on quality of the different design suggestions.

MEMIC may provide a list of workable suggestions, but such a tool may be enhanced by incorporating some indication of how *innovative* the suggestions are. By ranking design suggestions by innovativeness, the designer may focus more on trying to incorporate a component that has been shown to enhance the perceived innovativeness of other designs. The CARRIE algorithm develops a method to determine innovativeness at the component level, and adding this ranking functionality to MEMIC or other suggestions may improve its ability to promote the human generation of creative designs.

2.2.4 Related Work in Design Creativity and Innovation

The search for metrics for assessing the innovation of a design is not a new study, and is a source of recent research in the design community. Consumer reports and market success have formed a part of this work [7], though others would argue that this characteristic has sociological roots [12]. For a reasonably-scoped study we look at methodologies generating a innovation score for an isolated set of concepts rather than incorporating market or sociological trends.

There are many approaches to identifying innovativeness. Hirschman identifies the effect of innovation on the marketplace as: "If there were no such characteristic as innovativeness, consumer behavior would consist of a series of routinized buying responses to a static set of products. It is the inherent willingness of a consuming population to innovate that gives the marketplace its dynamic nature" [6]. The idea of 'innovativeness' is also closely related to 'creativity'. As Yusuf puts it, "Innovation springs from the creative application of knowledge" [25], indicating that creativity is actually the *precursor* to innovation. Thus we desire to promote creativity in the conceptual stage of design in order to increase the innovation in the final design. The difference between innovation and creativity in the literature become muddled, though we define 'creativity' as affecting the coneptual generation of ideas before the design, and 'innovation' as the applied ideas of creativity.

There has been a large focus on teaching creativity and innovation to design students, as these qualities have such a high bearing on product market success [24]. Metrics have been developed to automatically assess the creativity of the ideas produced by an individual design student [8]. More commonly there is a drive to provide new tools to designers that promote creativity during the early design process. These tools typically come in the form of computer software that either visualizes or supplements the information available to a designer.

The creative design process has been modeled in many ways, including evolutionary algorithms [11], co-evolutionary algorithms [5, 14], and genetic programming [10]. The evolutionary techniques provided in [11] were meant to guide human exploration in the design process through visualizations of design mutations. Human guidance was needed to tailor the genetic algorithm's solutions, and thus the algorithm was simple but did not fully encapsulate the creative design process. Maher uses the concept of co-evolution to

model the design process as a co-evolution problem [5], which provides a useful structure for the design process, but the fitness of this product is not identified. Koza actually develops an invention-creation system using genetic programming, and is able to verify the validity of this approach by having it 'invent' products and comparing them to inventions historically created by humans.

Quality, quantity, novelty, and variety were identified by Shah et al. [19] to evaluate a set of concepts. In particular, the metric of novelty has been implemented in recent work by Oman et al. [17] that attempts to identify creativity through product comparison. The metric for novelty we use in this paper is similar to the one used in Oman et al., and is given by the equation:

$$S_{Nj} = \frac{T_j - R_j}{T_j} \tag{2.3}$$

where S_{Nj} represents the novelty, T_j is the number of designs in the set being examined and R_j is the number of times the component solution for function j is seen in the set. This novelty metric has been traditionally applied only over sets of products with a similar type or goal, but in our work we use it over a set of products of varying types and goals.

Novelty, which purely relies on frequency with which a solution appears, is a simplistic approximation of innovation or creativity. As a simple counterpoint, a solution may appear infrequently because it is a one-time mistake by the company rather than any creative inspiration. Conversely, a component which appears *frequently* in a dataset to solve the same function may almost always be regarded as *not* innovative, giving back some validity to using novelty with creativity. For this reason, Oman et al. use this value combined with a weighting of perceived creative contribution to find the creativity of a product as a sum of these weighted novelties. We draw inspiration from this approach in our derivation of our difference reward, as we will discuss further in Section 4.

2.3 Statistical Analysis Methods

In order to identify innovation at the *product* level, we use a statistical evaluation of the survey that we gathered to assess this quality indirectly. Instead of directly polling for the assessment of the innovation of a product, we use a latent variable model, which related

quantifiable product attributes (i.e. patents, features, sustainability) to hidden attributes (i.e. innovation, product usability, company profile). We then use the score derived from this statistical method as a product-level score input to the CARRIE algorithm.

A 'latent' quality is one which is hidden, or cannot be directly observed. 'Latent' variables are qualities which are suspected to have some correlation with a measured quality, but the relationship between the measured quality and the latent variable cannot be directly defined. Innovation is such a quality: we believe it relates to the marketability of a device (which is measurable), but we cannot directly measure this innovation score.

Latent variables have been used in several applications to statistically characterize the impact of hard-to-identify characteristics. They have been particularly useful in the fields of medicine and psychology [4, 23]. *Latent variable theory* in the psychology community is the theory "in which a mental disorder is viewed as a latent variable that *causes* a constellation of symptoms" [13]. Latent variables have also been used to determine correlations between different brain functions while performing certain targeted tasks [15].

A latent variable approach uses a statistical regression to determine the impact of a latent variable on a measurable score. Practically, this is taken care of for us by the Stata 12.1 software. Latent variables are useful because of two main qualities: they can be used to solidify observations in data using a small sample set, and they do not force correlation. Because we tend to have a statistically small sample size for much of our data and we do not want to *assume* that the innovation has any influence on the marketability, this is a good technique to employ. We use an innovation latent variable as a product-level score.

The human factor is not removed by latent variables. The latent variable technique simply functions as a method to identify correlations between calculable properties of an object and a hidden element (the 'latent' variable). Once this correlation is found, the human element *can* be removed, and the latent variable may be calculated through measurable factors in the design. This may be a good technique to expand the learning dataset in the future, but this is not explored in this work.

Chapter 3: Algorithm Overview

In this chapter we examine in detail the CARRIE algorithm. We first provide a high-level overview of the CARRIE algorithm as well as the process by which it receives training data. In Section 3.3 we analyze the structure of our multiagent system and how it takes in data, makes forced training selections, and receives a reward. The process by which the rewards are redistributed and given to the agents for reward updating is shown in Section 3.5. The reward that the agents receive is also derived in Section 4. A brief explanation of how the learned data are used is provided in Section 5.1.

Figure 3.1 shows the general process that the CARRIE algorithm uses. This process will be examined further in Sections 3.3 through 3.5. A chart showing the flow of the full algorithm in detail is also available in Appendix A.

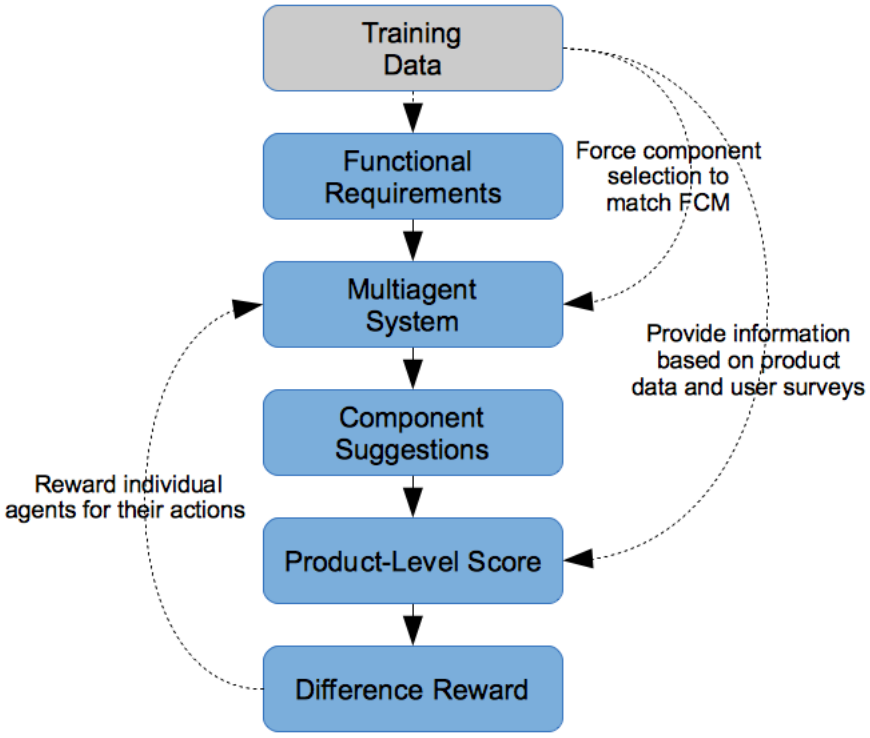

Figure 3.1: A high-level view of the CARRIE algorithm. Functional requirements are taken from the training data, and then given to the multiagent system, which then outputs a selection of components. This selection is guided by the components found in a preexisting design. The 'design' offered by the multiagent system is then scored as a whole, which is obtained from the innovation score of the device as a whole. This is then turned into a difference reward for each agent who participated in creating the design, and this difference reward is then fed back to the multiagent system. Gray indicates that the training data are not modified in this algorithm, and dashed lines denote information transfer from the training data to the CARRIE algorithm.

The data shown in gray and in dashed lines are inputs to the CARRIE algorithm. The training data are used first to acquire the functional requirements necessary to replicate a given design. These functional requirements are used in the multiagent system to call the agents which participate in the design process. For training, the multiagent system

also takes in the data from the FCM to force the agents to select components which are consistent with the FCM. The multiagent system then passes on the component suggestions which, in training, form the original design passed in. The product-level score is then calculated from (or directly given by) information from the training data. This score is turned into a reward and used to reward the participating agents for their respective actions.

Here we describe the training usage of the CARRIE algorithm. The same general framework may be used if the system is used open-loop to provide design suggestions to a user. The user simply has to input functional requirements, then receive the most innovative suggestion based on the pre-trained data given by the value tables of the agents identified by the functional requirements.

3.1 Training Data

We train using data from the Design Repository using the process shown in Figure 3.2. For each training episode, a design is selected from the repository. Two things are then acquired from the design; the function-component matrix, and information about the design which will enable calculation of the product-level score (this calculation process is external, and discussed further in Section 3.4). The FCM is then used for two things. A list of functional requirements are taken from the FCM, identifying and tabulating which functions are required to create the design. Later in the algorithm, the multiagent system is constrained to selecting components in the FCM, effectively 'copying' the component selection made by the original designer.

These training data are separated from the CARRIE algorithm because the Design Repository, although useful, does not necessarily need play a role in this process. For example, a customer looking for design help might just directly give a set of functional requirements to the CARRIE algorithm, get a set of components used to build a product which have maximum innovation according to the multiagent system, create a product based on those suggestions, and then get a product-level score from a manager seeing the new design. The CARRIE algorithm merely requires a set of functional requirements, a scoring mechanism, and some way to assemble the components given (or constrain them to the components found in an existing product).

There is another reason that we separate out the training data from the CARRIE

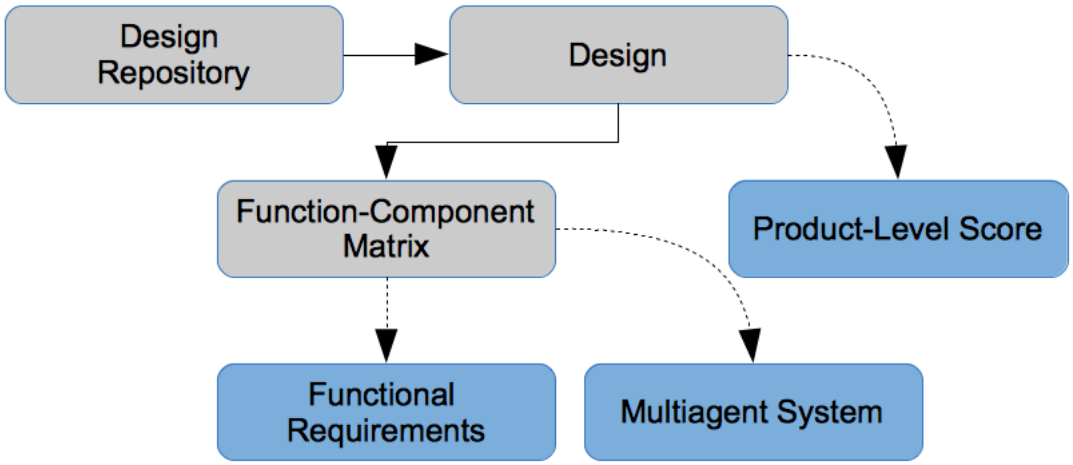

Figure 3.2: The training data and how it fits into the rest of the (simplified) CARRIE algorithm. Product-level scores are obtained using information from the design information in the repository, and the multiagent system uses data from the function-component matrix to restrict component selection. Functional requirements are also taken from this function-component matrix by identifying which functions appear in the design.

algorithm; we want to make our algorithm expansible. At this time we must use human-gathered data to assess innovation at a product level, but in the future there may be a better product-level scoring mechanism. This might be added on to the CARRIE algorithm or it might be kept as an input. Alternatively there may be surveys with more resources and less bias which provide the CARRIE algorithm with a better set to train from.

3.2 Agent Definition

Although this has been mentioned briefly in previous sections we seek to clarify the role of an agent at this point. It is crucial to the understanding of the rest of the CARRIE algorithm that the reader be familiar with the agent's learning mechanism, its limitations, and its conceptual purpose within this system.

Agents have unique functions assigned to them. Their task is to provide design suggestions which, in conjunction with the other agents in the system, increase the product-

level innovation score of the design. By performing action-value update as described in Equation 2.1, the agents learn an internal evaluation of how their component selection generally improves the innovation of a product.

During modeling of the system we could have chosen a number of possible formulations for the agents. For example we could have flipped the agent with its action selection and had components define an agent; the role that a component played within a product (its *function* choice) would then be evaluated, and the mean value of its Q-table may have been determined as its full score. Alternatively agents could have been formulated as designs; the design would select components and then receive a product-level score for them regardless of functionality.

We chose our agent formulation based on the data available to us, as well as the idea behind a multiagent system. In order to have a multiagent system, we must define elements which interact to contribute to the product score, and whose actions are directly tied to the product score. This eliminates the potential of framing the designs as agents, as they do not interact with one another. The other formulation, where the components select a functional role in a product, is more reasonable. The components do interact directly with one another, and we assume that their functionality within the product has some impact on the product's innovation. This is a convenient formulation, but lacks a real-world analogue. We approached this from a design perspective: a designer is seldom given a list of components and told to select the way to use them. More often, a function is required by the abstract model of the design, and the task of the engineer is to select the best component that performs that function. For this reason, assigning the agents functions to satisfy better emulated this process. Additionally, it allowed us a method to use this information in much the same manner: if a query is made to the agent to fill a specific function, the agent will be able to return a component or set of components which may be used to perform this function. Because the end goal of this work is to improve the concept generation methods, we selected agents which could take the same type of inputs as other concept generation software.

3.3 The CARRIE Algorithm

In this section, we describe our multiagent system, and how it trains on data from the system. The multiagent system is the mechanism by which we score the different

components in the dataset. Each agent in the system keeps an internal table of values corresponding to each action (component selection) available to it. This is key, because the agent's evaluation of the component selection is what we are interested in learning.

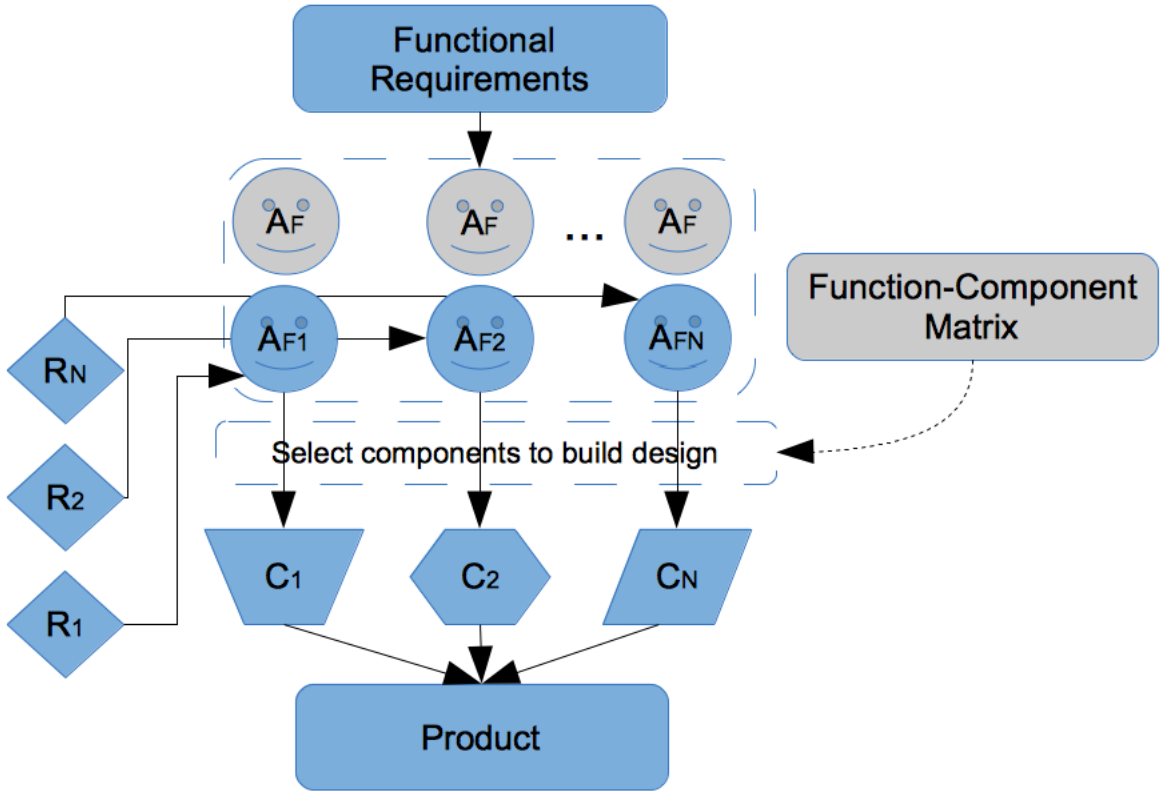

Figure 3.3: The information flow in and out of the multiagent system in the CARRIE algorithm. Functional requirements are given to an agent selection process. The selected agents then select design components. These are constricted during training to the components found in the training set design. The difference reward is then given to the agents after the evaluation process.

The multiagent system takes three inputs from the rest of the system; a set of functional requirements, the FCM for the training device, and a difference reward. The set of functional requirements is required as an input because when training from each design, we require only a *subset* of the agent population. Not every product has every function

in the set, and these functional requirements are used to identify which of the agents actually play a role when training from an example design.

We cannot obtain a product-level innovation score for an arbitrary product. Scorings are only available for a select subset of the design repository, and therefore we cannot allow agents to select actions freely or there will be no method of coming up with a reward for these actions. For this reason, we use an FCM to force the agents to select actions shown in a training design (for which we *do* have a score). The agents thus essentially recreate the function-component matrix, simulating making the design decisions to select the components given in the FCM. This FCM is then output as a 'product' which has an associated score.

The agents receive a difference reward for taking each action, and their evaluation of their action is adjusted according to the update equation given by Equation 2.1 with an α value of 0.5. An agent stores this information so that it can iterate upon it with later training data. It is this adaptability that makes this a flexible approach to component evaluation; if more data become available, existing component scores can be updated periodically to reflect this new data. This process is also somewhat sensitive to the order in which the training data are presented. If we had a large dataset we could set a lower learning rate and assume that convergence would occur after some amount of noise, but our dataset is too small to assume that we will converge immediately to the true value of the component. To mitigate this variation in introduction order, we introduce training designs to the component in a randomized order during the learning process, and take the average component score over statistical runs.

The main idea behind our multiagent system is to model the *design engineering* process as a system of agents seeking to fill functional requirements by selecting components. Our agents essentially act as a team of engineers, working together on their own subsection of the design product to make the most innovative product possible. It is perhaps more appropriate to call the agents 'engineers-in-training', as they are actually learning from the master engineers who originally designed the products in the repository.

The structure of this multiagent system and reward process is key to this work because it allows a learning agent to gain design experience, which can then be used to assist designers in developing innovative design suggestions. Design choices are generally made using the experience of the engineer, but gaining design experience may take years and only cover a small subset of designs. By automating this process, we can transfer a

summation of the 'experience' within the repository and our product-level score set to the engineer. This is the output of the CARRIE algorithm.

3.4 Product-Level Scoring

The innovation score cannot at this time, from the data available in the Design Repository, be objectively calculated for a product. Luckily, innovation is a quality that humans can readily assess. Consumer magazines periodically publish lists of top innovative designs, and people can often identify an innovative product from a set of products. We leverage this fact by scoring our products in three ways: a binary score given by the data from experts, a score from the average value of survey data, and a score given by an innnovation latent variable calculated from a survey of design engineering students.

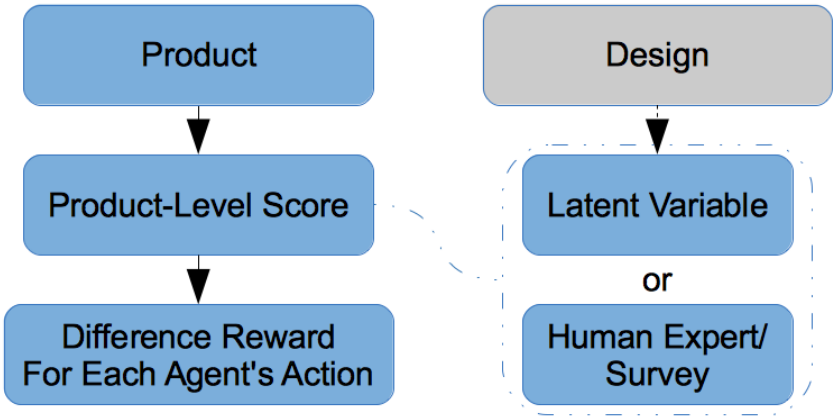

Figure 3.4: General product-level scoring method. A product is analyzed to produce a product-level score, which in our case is determined from the Repository's design information. Depending on the assessment method, this is then subjected to review by a human expert, or information is gathered via a survey and either a latent variable or a mean value of responses is used. This product-level score is then fed to the difference reward calculation, which turns it into an agent reward.

Figure 3.4 describes how this data plays into the multiagent system. We use the data from all three sets as an input to the difference reward calculation ($G(z)$, as given by Equation 2.2). The $G(z_{-i} + c_i)$ term is actually calculated from this score as well, but this will be explained further in Section 4.

There are three different ways that we gather data on the innovation of different devices: expert innovation identifications found in consumer magazines, a survey of several college students, and a latent variable analysis using data from design engineering students. These are explained here in further detail.

Expert Data: The first approach at obtaining product-level evaluation of designs was to look to the experts: *Popular Science*, the IDSA IDEA Award, and *Time Magazine*'s 50 Best Innovations had, in a collaborator's work, been used to identify innovative designs which were available within the repository, and to compare the functions found within these designs to the functions within a set of comparatively non-innovative designs. Though their approach was to identify *functionality* which added to the innovation of the designs, the identification of innovative designs nonetheless provided a good testing dataset for the CARRIE algorithm to identify innovation at the component level. The data we used as a product-level score may be found in Appendix B.

Survey Data: To make up for the unnatural discretization and small size of the expert dataset, an informal survey was conducted involving 10 participants who were assigned to rate a series of 50 designs in a scale of 1-5, 1 being the least innovative and 5 being the most innovative. The average of this data was then taken, and this was used as a product-level score for the designs. This was then scaled between 0 and 1 for comparison with the expert data. This averaged and scaled score may be found in Appendix C.

Latent Variable Data: The main weakness of our dataset has been its small size or lack of detail. With the expert data we may be confident in a correct assessment of our product's innovativeness, but we are not told *how innovative* it is. Conversely, with our survey we obtain how innovative a product is, but the survey-takers have varying backgrounds and there is significant variance in the data which is exacerbated by the small sample size. Latent variables provide a statistical technique to overcome this problem of small sample size. A collaborator performed a latent variable analysis on the data which involved innovation as a latent variable across 8 products. The values for the innovation latent variable was used as a product-level innovation score, and the data for that may be found in Appendix D.

The training data were gathered from human sources, and therefore necessarily had

some variation and bias. The methods of collection were not only different, but also the populations from which the data was collected were different.

Each dataset had a different method of collection. The expert data was collected by gathering a set of innovative products identified by *Popular Science*, *Time Magazine*, and the IDSA IDEA award and purchasing a selection of these innovative products for reverse-engineering and addition to the design repository. Thus, there was a financial bias in the data, in that there was not an unlimited amount of money to acquire the innovative designs, and a design engineering graduate student was responsible for the acquisition of this subset. Additionally, the comparison products were also identified by a design engineering graduate student.

The survey data was also performed on a different demographic. The survey population in this case was a mix of four graduate mechanical engineering students, three graduate computer science students, a graduate design and human environment student, an undergraduate history student, and an undergraduate environmental engineering student. This set consisted of three women and seven men. There was no given definition for the 'creativity' that they were scoring in the designs shown to them, and thus they had to rely on only personal bias to formulate a scoring. A maximum time of one minute was given to assess the creativity value of a product which was shown with a picture and a brief explanation of the how the design was used.

The latent variable analysis was perhaps the most targeted at reducing variability induced by the dataset, but it still targeted a specific demographic. The entire dataset of 156 responses consisted of undergraduate engineering students who were taking a mechanical engineering design class. Each student responded to a set of questions which were targeted at identifying the impacts of three latent variables: innovation, product usability, and company profile. Because the process of assessing latent variables makes room for the fact that some of the latent variables may in fact have no impact on the final score of the component, the presence of the other latent variables did not necessarily have a direct affect on the scoring of innovation, but the background of the engineering students would impact their answers to the survey questions, and thus propagate down to the results. This is not necessarily a bad thing, as the demographic in this data is relatively controlled, but it does limit the applicability of the findings using this data to other markets.

The data used to assess product-level scores may have influences from the demo-

graphics surveyed and the designs targeted, but the dataset we obtained still has value in that it is internally consistent. Furthermore, innovation is not an objective concept. Asking a team of market analysts, a mixed set of college students, and a set of design engineering students to define innovation, will result in different definitions of innovation. These definitions may have less variability within their specific demographic, however, and thus we can still find trends in this data.

3.5 Reward Feedback

Reward feedback is given to an agent so that it can add to its 'experience'. Rewards give information to the agent about whether the action it took was bad or good, and *how* bad or good its action was. In systems where agents are using their policy to make decisions, their action selection depends on the rewards which they have seen previously. Though we do not make decisions based on the policies of the agents, the information on the ranking of the actions is what we hope to obtain from the CARRIE algorithm, and therefore it is important that the reward be reflective of the agent's true contribution to the system.

The general method of our reward feedback as used by the CARRIE algorithm is given in Figure 3.5. The product-level score, which is derived from the agent's component selections, is fed into our difference reward which is used for decomposition of the reward. The specifics of this reward are explained in Section 4, but it primarily functions to break the product-level score down in a way that diminishes the noise from the actions of other agents in the learning signal. This difference reward is then fed back to the agents, rewarding their action selection so that they can update their policies.

This reward feedback is an essential process in reinforcement learning. It allows agents to adapt to new and changing data. We are using agents as a tool, so we don't need them to necessarily *adapt* so much as *simulate adapting* to the data. This allows for an agent to adjust the estimation process as new data become available.

An agent learns over just a few training examples–in the latent variable dataset there are as few as 8 products on which it trains, and the survey dataset has the most data at 50 products. This is a small number for learning to occur, and therefore we must take advantage of reward shaping techniques such as the difference reward to speed up the convergence to good policy.

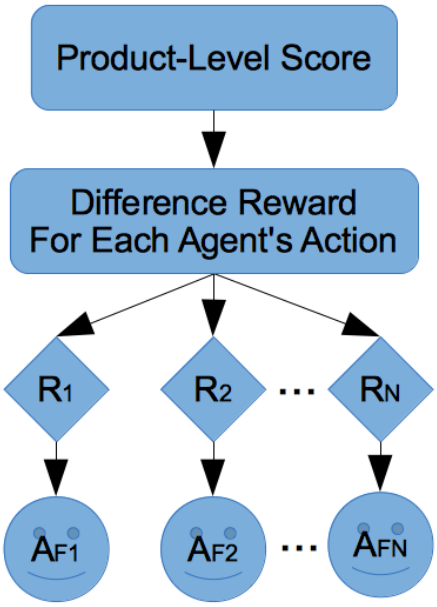

Figure 3.5: Diagram of how the difference reward interacts with the product-level score and the multiagent system. Difference rewards are computed for each agent and then given to the agents for incorporation using the value-update equation.

Chapter 4: Difference Rewards for Component Scoring

In Chapter 3 we developed a multiagent system which used designs from the Design Repository as training data, and received an external evaluation of the product's innovaton. This innovation evaluation represents a black-box score, and applies to the entire multiagent system; agents cannot observe how this score was formulated, and therefore do not know how they contributed to that score.

We can use difference rewards to modify the system-level score to determine how each of the agents contributed to a score, but the difference reward formulation given in Equation 2.2 requires the functional form of the system-level score to be known. However in many domains it is difficult or impossible to calculate the effects of removing an agent directly from the system-level reward. In these cases it has been shown that approximations of the difference reward can still take advantage of the noise reduction from this reward shaping approach [18]. Though we cannot compare to a product which has been redesigned without a component choice, we can approximate the effect of a component's removal using a metric which has been used in previous work as a rough approximation of innovation; the *novelty* score.

In this section, we derive a difference reward approximation by taking the product-level score, which represents the $G(p)$ value in the difference reward, and then using a novelty-based approximation of the $G(p)$ function to determine the value of the product-level score in which the agent does not participate in the design.

4.1 Derivation of Difference Rewards

To use the difference reward as given by Equation 2.2, we need an equation for the system score of a device which we can modify to mathematically remove the impact of an agent. We assume that we begin with a product-level evaluation of the innovation $G(p)$, which we obtain for our experimental results using any of the three methods described in Section 3.4. Similar to the work in Oman et al. [17], we assume the innovation score is composed of weighted novelties. We also assume that each component contributes to the

score proportionally to its novelty, and we weight these novelties based on the system score over the sum of novelties:

$$G(p) = \sum_{i=1}^{N} S_i \cdot \frac{G(p)}{S_{sum}} \tag{4.1}$$

where i refers to the i'th component found in a design, N is the number of components found in the design, S_{sum} is the summation of the novelties, and S_i is the novelty score of that component. For our difference derivation, we assume that this weighting term ($\frac{G(p)}{S_{sum}}$) is *static*, and therefore it does not change when the difference reward is applied, despite the fact that it involves the effects of the agent. We call the system reward without a component j with a counterfactual:

$$G(p_{-j} + c_j) = (\sum_{i=1, i!=j}^{N} S_i + S_{cj}) \cdot \frac{G(p)}{S_{sum}} \tag{4.2}$$

Which gives us the difference reward:

$$D_j(p) = G(p) - G(p_{-j} + c_j) \tag{4.3}$$

$$D_j(p) = G(p) - G(p_{-j} + c_j) = \sum_{i=1}^{N} S_i \cdot \frac{G(p)}{S_{sum}} - (\sum_{i=1, i!=j}^{N} S_i + S_{cj}) \cdot \frac{G(p)}{S_{sum}} \tag{4.4}$$

$$D_j(p) = (\sum_{i=1}^{N} S_i - \sum_{i=1, i!=j}^{N} S_i - S_{cj}) \cdot \frac{G(p)}{S_{sum}} \tag{4.5}$$

$$D_j(p) = (S_j - S_{cj}) \cdot \frac{G(p)}{S_{sum}} \tag{4.6}$$

where $D_j(p)$ is the difference reward for agent i learning on product p, S_j is the novelty score of the component which agent j selected, S_{cj} is the counterfactual novelty (discussed in the next section).

4.2 Implementation of Difference Rewards

We use equation 4.6 in simulation to give the agents for their action selections. This equation leaves one selection out; the counterfactual novelty score that is used. We propose two counterfactual values for this difference reward. The first uses the lowest novelty score, which is similar to previous work in which a 'common' product was functionally compared to an 'innovate' product. The second uses the average novelty score within the device, which essentially holds the agents to a higher standard. This indicates that if an agent is below average, it is actually *detracting* from the design's innovation score.

The difference reward has a counterfactual component, meant to add in a 'substitute' action for the replaced agent. In this work it is logical to have a nonzero counterfactual. For example, a toaster minus its heating coil would simply be a broken toaster. A toaster with its heating coil replaced by a propane torch might still accomplish the original task output by the device, but would not necessarily have the same innovation score as the original toaster.

Previous work by Oman et al. [17] has sought to emulate difference rewards by comparing a design having an innovative component or functionality to a design which had standard components or functionality. They found that they could identify functionality as 'innovative' using this method. Our objective is different; we aim to identify innovative *component* solutions rather than added functionality. For us to compare and contrast what is 'innovative' and what is 'common' requires that 1) there be a design in the repository which has the exact functionality and components of a more 'innovative' device except for the functionality which we are scoring and 2) we are able to even identify a 'common' design to contrast with.

In the absence of a team of engineers and market experts willing to continuously reconstruct and re-score devices so we can use our difference reward, we turn to two heuristic methods. The first is inspired by the work by Oman et al., and aims to compare a component found in a product to a 'common' component, which is set as the counterfactual. For this we must define the concept of 'common' as a component that has the lowest novelty score, $S_{j,least}$, which still is found within the set to solve the function in question. We get the difference reward:

$$D_j^{least}(p) = (S_j - S_{j,least}) \cdot \frac{G(p)}{S_{sum}} \tag{4.7}$$

But what if we don't compare to the lowest-scored component–what if we compare to just an average component? This may give a better idea of whether the component is helping or hurting the design because the feedback may be negative as well. If we compare to the average score, we get the difference reward:

$$D_j^{average}(p) = (S_j - S_{j,average}) \cdot \frac{G(p)}{S_{sum}} \tag{4.8}$$

If, according to our novelty-based breakdown of the score, an average component could have improved the novelty, the component is identified as *detracting* from the design and gets a negative score. Equation 4.7 will only return a zero or positive score, while Equation 4.8 will roughly half the time give a negative score. These two methods of assessing innovative impact give different but equally valid ways of assessing a component's innovation contribution.

Chapter 5: Results

Results generated with the CARRIE algorithm using the Design Repository Data are divided by the external evaluation method used. These varied both in products evaluated (subsets of the Design Repository) and the type of innovation information gathered (binary, from 1-5, or unscaled), as well as the population providing this data. In this chapter we first give an explanation of how these scorings might be used in Section 5.1, and then in Sections 5.2-5.4 we show the data gathered and identify some of the trends found. In **??** we briefly analyze the trends found across all datasets.

5.1 Use of Aggregated Data

How do you use data gathered by a multiagent system? This is a driving force behind this work, and it is important to make this clear so that this data may be used in future work. We have set up a system in which agents may work together in a multiagent system to simulate the design process. This multiagent system then has a ranking for component within the design set divided by the function they perform (agent to which they were assigned).

Value update usefully catalogs information about innovation of components within the agent's *policy*. Each agent keeps an internal score for each of the components within its policy. To use the data collected by the agents, the policy of the agents must be examined. The CARRIE algorithm has the capability to output the policies of the agents. The same components may be found in the policies of different agents, but they have different scores due to the fact that they perform a different function within the device. These policies are graphed in Section 5, and each value table score is plotted against the averaged product-level score for a function-component pair.

The aggregated policies acquired by the multiagent system can be used in three ways; the creation of an automatic design-generator, the development of a component ranking system, and as a method to identify the trends in innovativenes. To have an automatic design-generator we simply need to first train our agents on existing innovation data and

then 'take off the training wheels' – instead of having component selection be forced, we simply need to give a functional requirement set and a selection heuristic (epsilon-greedy or softmax would suffice), remove the reward feedback, and obtain the component selections from the multiagent system.

Retrieving innovation scorings is an even simpler process. We must simply look at the policies of the agents in the multiagent systems to acquire their assessments of each component's innovativeness in solving a particular function. Once this is retrieved, it can be used as a ranking system for an external concept generator such as MEMIC.

We focus primarily on the third method of using this data; analyzing trends in innovation as it relates to the average score of a device. Part of this is to test the effectiveness of the CARRIE algorithm to make sure that it propagates the innovation scores in a manner in which we expect. There is no gold standard in the learning or design literature for identifying innovation at the component level, so we cannot know how *accurate* we are in learning this data, but we can identify trends from using these techniques.

We have two difference reward formulations (D^{least} and $D^{average}$) and three datasets which we can work from (expert, survey, and latent variable), so we begin by running six different experiments exploring these parameters. Reinforcement learning depends on the order in which different rewards are given (earlier rewards have more emphasis than later rewards), and we run experiments using a randomized order and take the average across 30 runs.

5.2 Expert Evaluation Dataset

As a baseline we began, as shown in Figure 5.1, by starting with the most well-established method of identifying uniqueness in a dataset; the novelty. We plotted the novelty of function-component pairs versus the average product-level scores in which the function-component pair was found. Simply plotting the novelty against this average score served to solidify a key hypothesis in the beginning of this work; *the frequency with which the component appears does not solely reflect the creative impact it has.* Novelty scores tended to be higher at the extremes of the average product-level score, with slightly lower values toward the center of the spectrum, but this correlation was weak in this data.

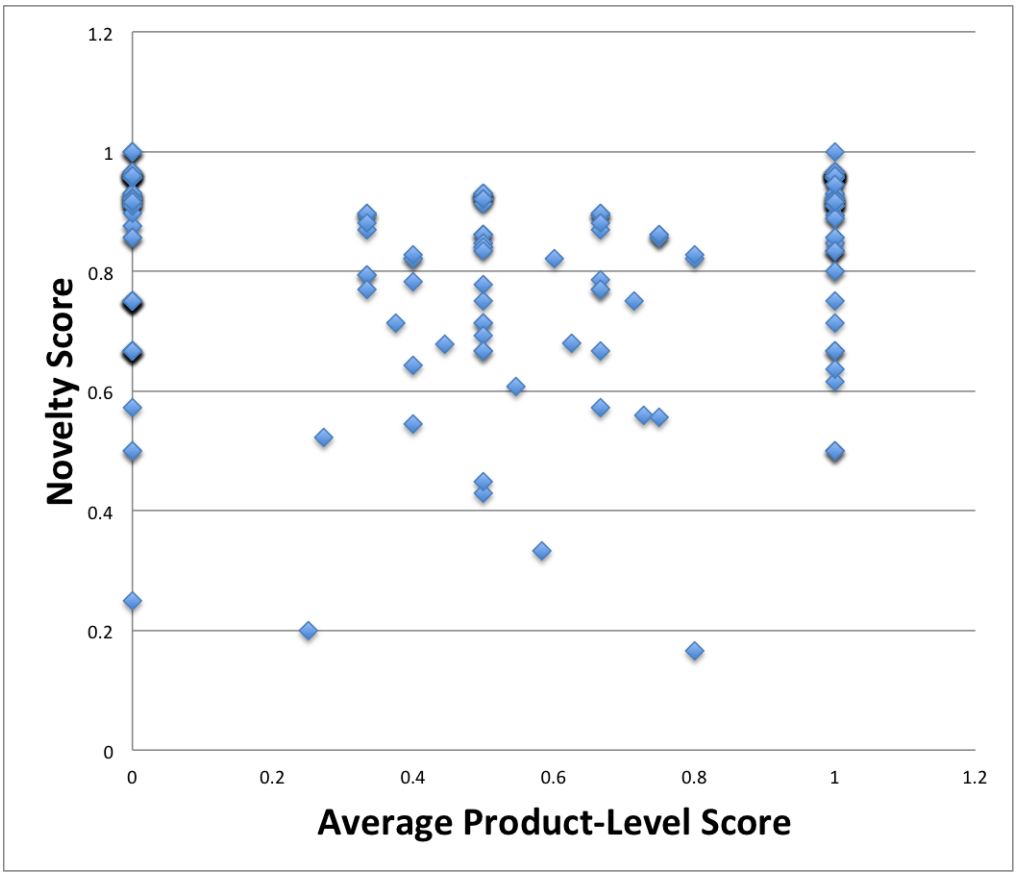

Figure 5.1: Novelty scores for the function-component pairs found within the expert evaluation dataset. The data show no clear trends apart from higher novelty scores at the extreme average product-level scores.

Figure 5.2 shows the results from our multiagent system method of evaluating the innovation at the component level. Innovation scores showed a general increase with average product-level scores, but also showed a marked increase in spread as the product-level average increased.

The interesting part of this data is not necessarily in the trends, which will tend to increase with the average product-level score because of the learning mechanism. The outliers give a better indication of how a component might be cleverly used to perform some function in the dataset. In Figure 5.2 we see two major outliers: the datapoints

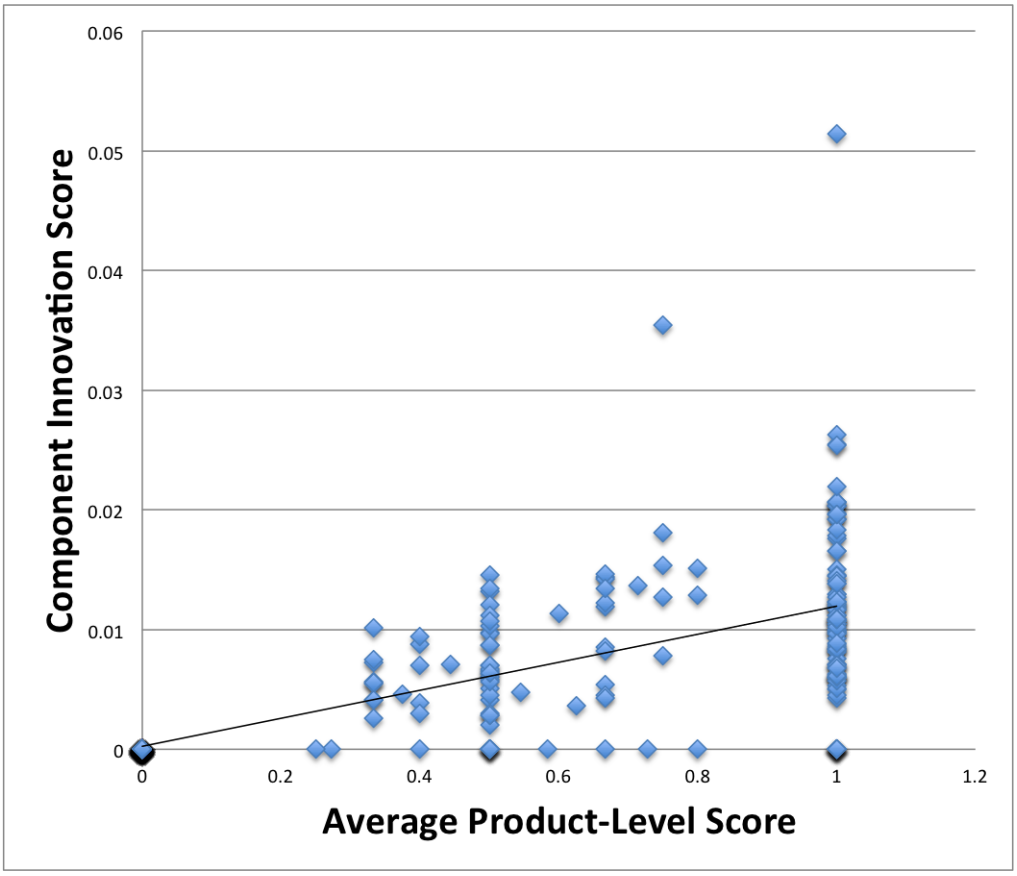

Figure 5.2: Agent-estimated values using the D^{least} reward on the expert evaluation dataset. The data show an upward trend with two major outliers. The outlier shown at (0.750, 0.035) is the (channel material,reservoir) function-component pair, while the outlier shown at (1.000, 0.051) is the (control magnitude material, screw) function-component pair. Both of these show particularly high estimations of contribution to the innovation score.

corresponding to (channel material, reservoir), and (control magnitude material, screw). While these components do not appear innovative in and of themselves, they might serve a particular function within their design in an innovative way.

Figure 5.3 shows much less correlation with the average product-level score than Figure 5.2. This is likely due to the fact that the $D^{average}$ difference reward has an average-novelty counterfactual, and therefore will tend to have an equal number of points

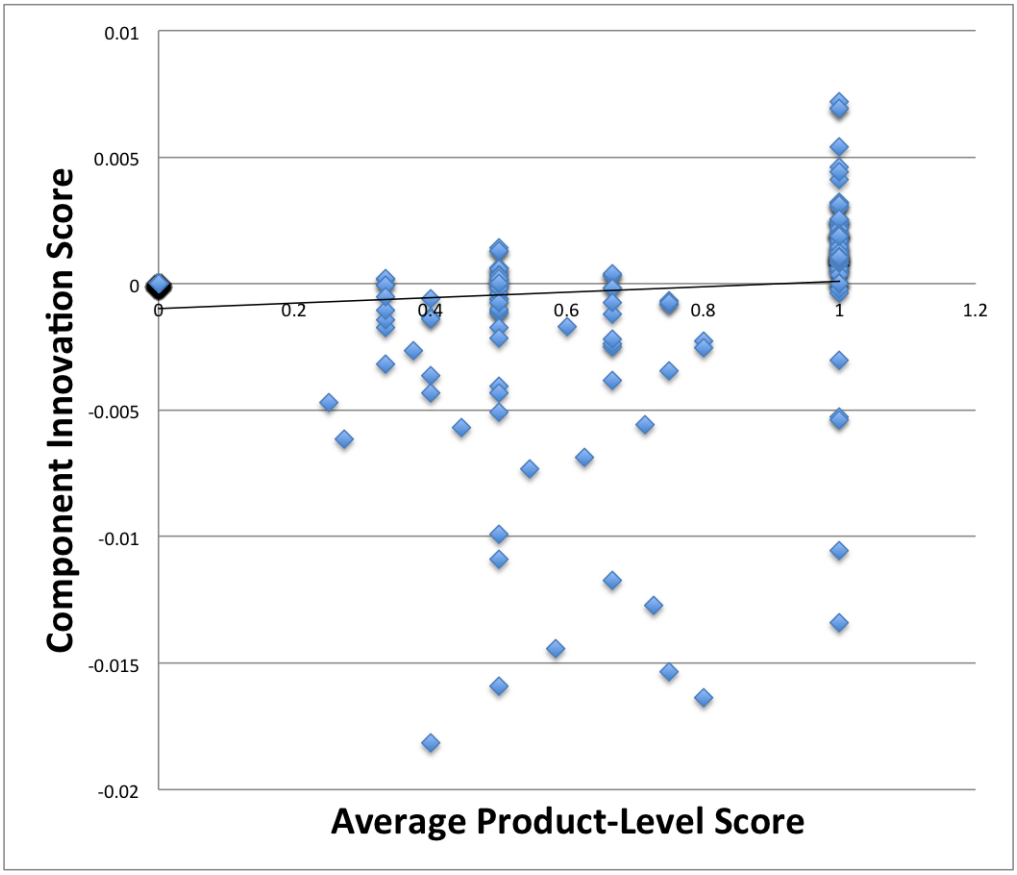

Figure 5.3: Agent-estimated values using the $D^{average}$ reward on the expert evaluation dataset. The data show a slight upward trend with have a wide variation in values. Negative innovation values are estimated by the agents due to the fact that $D^{average}$ uses an average-novelty counterfactual rather than a lowest-novelty counterfactual.

which are positive and negative. Most of the positive data are collected at the higher end of the average product-level scores.

Though the expert evaluation gave a good indication of how we could correlate innovation scores at the product level to innovation scores at the component level, the dataset upon which it operated was somewhat lacking. For one, the data was binary, and there were marked discretizations of the data which made it difficult to find some continuous relationship. Additionally, the dataset was relatively small for the lack of detail it offered: there were only 29 products in the dataset, offering little overlap between

functionalities.

5.3 Survey Average Dataset

The novelty scores for this dataset (Figure 5.4 were of some interest because they followed a similar pattern of having relatively high novelty scores at the extreme values of average product-level score, accompanied by lower levels of novelty at the lower product-level scores. The distribution of the data suggested that there may have been *more* data in the middle of the distribution, but it remains that, apart from a couple of outliers, the novelty of components both in highly-innovative and fairly plain products tend to be higher. Conversely, there seems to be a loose trend that the novelty of components at the lower-to-middle range of the average product-level score tend to be *lower*, indicating that products made from frequently-seen components may have a lower product-level score on average.

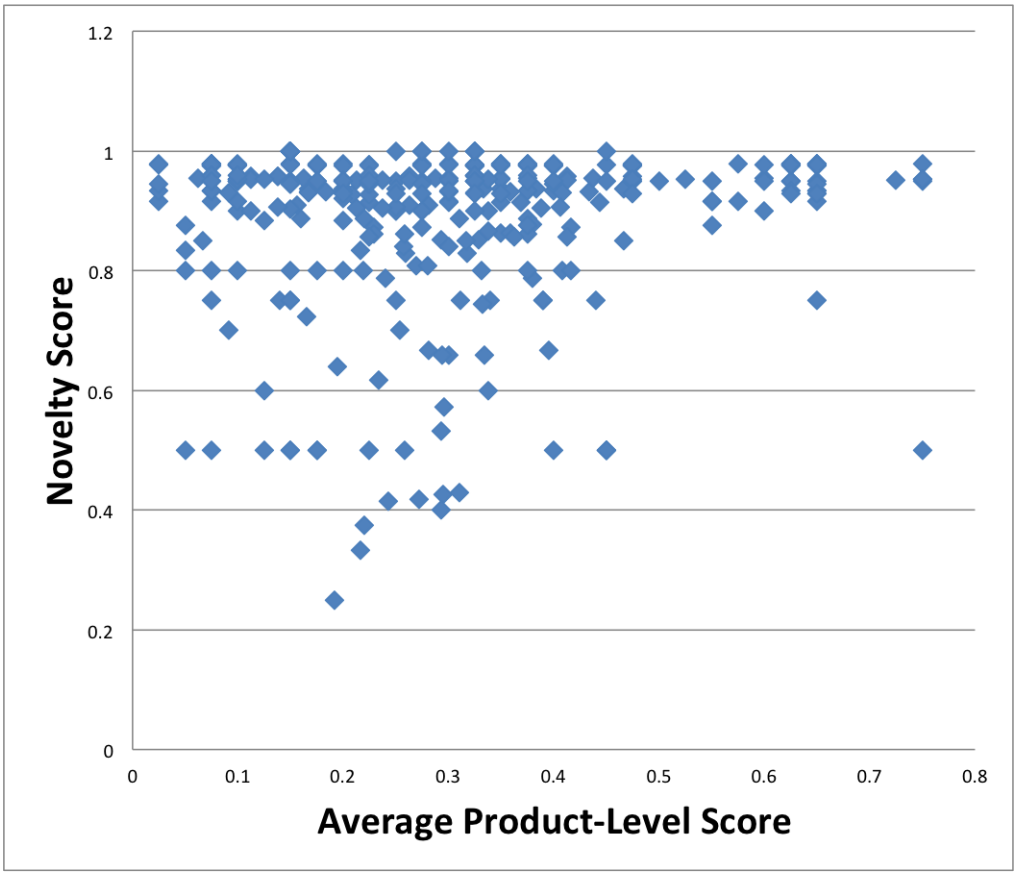

Figure 5.4: Novelty scores for the function-component pairs found within the survey dataset. The data show the diversity of the scores found in this dataset, as there is much less striation in the results as compared to the expert evaluation dataset. The novelty shows a general trend toward being high, but shows a dip on the range [0.05, 0.45].

The innovation scores shown in Figure 5.5 show a clear trend; as product-level score increases, the component-level innovation both tends to increase and polarize. At the lower levels of average product-level innovation, there tends to be a tight fit to the data. The variance in the data appears to increase with the average product-level score. At an average product-level score of 0.4, we see a dataset which disperses almost completely.

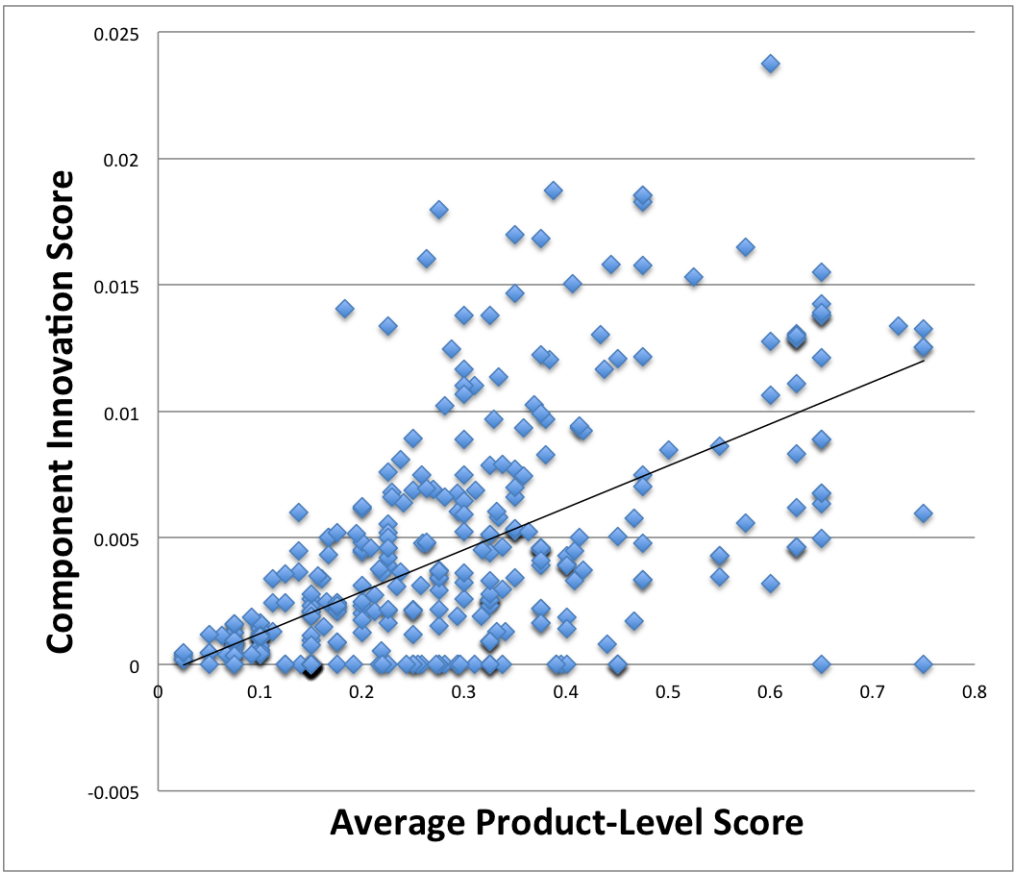

Figure 5.5: Agent-estimated values using the D^{least} reward with the survey dataset. The data show an strongly upward trend with high dispersal toward the higher ranges of average product-level innovation scores. One particularly highly-estimated function-component pair at (0.600, 0.024) corresponds to the (convert energy, handle) function-component pair.

The innovation scores shown in Figure 5.6 once again demonstrate the fact that roughly half of the function-component scores will score negatively. A trend which correlates somewhat with the novelty scores for this dataset (Figure 5.4) is shown in the data; function-component pairs which appear at the mid-level of average product-level innovation tend to *detract* from the design of a component according to our estimate rather than help it.

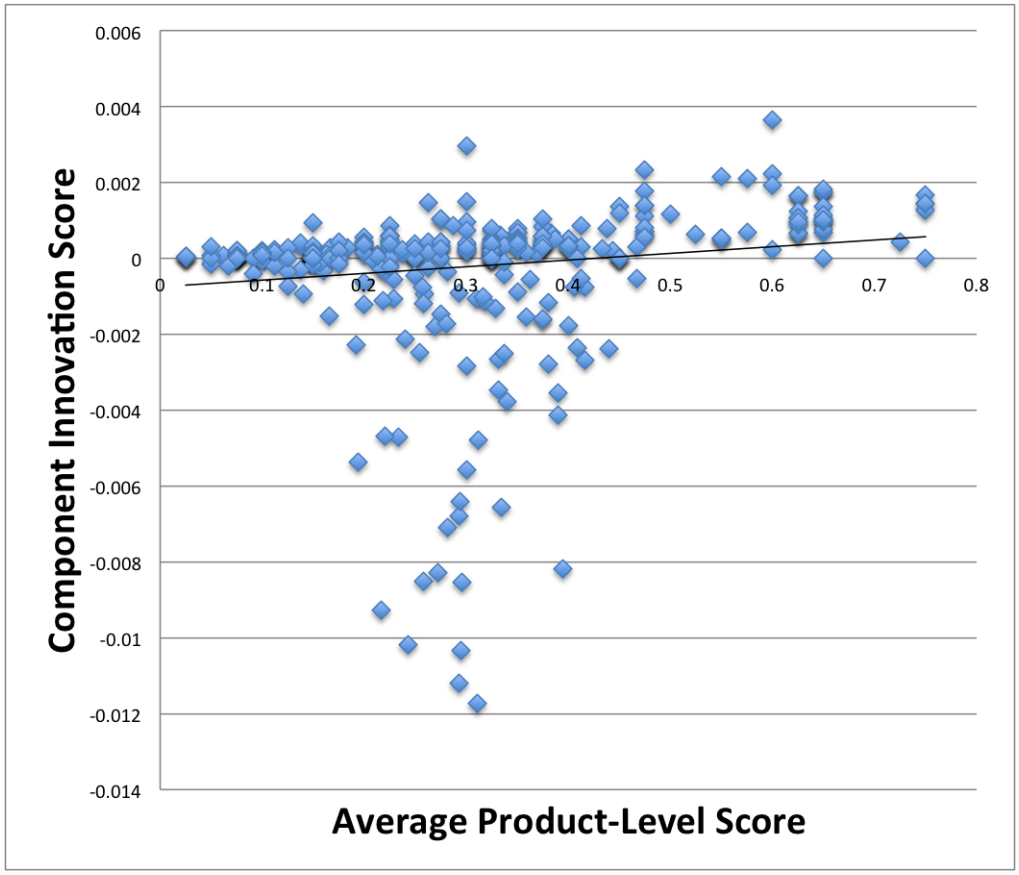

Figure 5.6: Agent-estimated values using the $D^{average}$ reward with the survey dataset. Though the data shows an upward trend, there is a large number of negatively-scored function-component pairs on the interval [0.15, 0.40].

5.4 Latent Variable Dataset

The latent variable dataset provides us a unique chance to test the CARRIE algorithm's response to negative product-level values. Scores in both our expert evaluation dataset as well as our survey dataset have provided a training set which is bounded on the interval [0,1]. We could have preprocessed the results to avoid negative scorings, but we wanted to test the robustness of the CARRIE algorithm and the reward structure.

As shown in Figure 5.7 the novelty within this dataset had the most pronounced

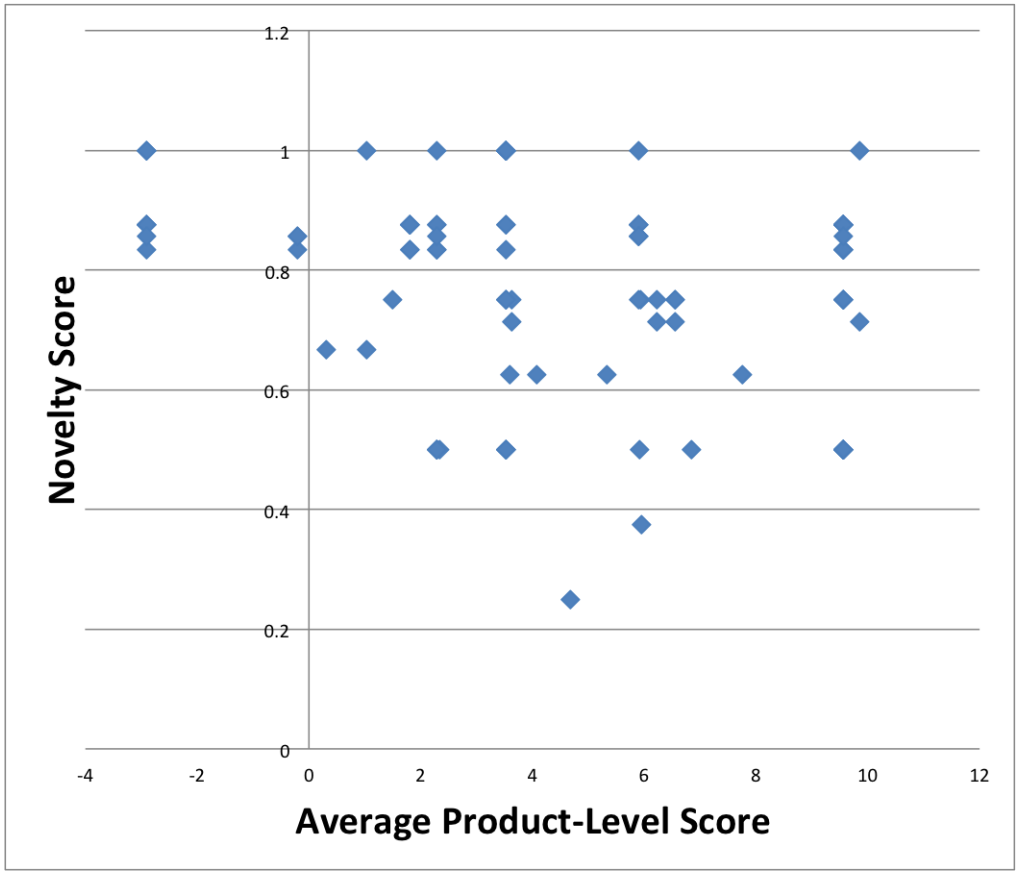

Figure 5.7: Novelty scores for the function-component pairs found within the latent variable dataset. Though the data are sparse, they are consistent with the novelties found in the other data in that they show a dip in novelty at the mid-level range. Strangely, even at the negative average product-level score, the novelty of all the function-component pairs are high.

trend in having higher novelties at the extrema and lower novelty scores toward the center. This data was almost triangular in shape, and further supports the trend across all datasets that novelty correlates to extreme high or low average product-level score. This is particularly pronounced because all negative average product-level scores have a relatively high novelty. Figure 5.7 also shows a certain amount of discretization which also exists in the expert data. The data are not striated so much as simply sparse, an therefore these discretizations are likely less due to the scoring mechanism and more due

to the fact that only 8 products are included in the analysis.

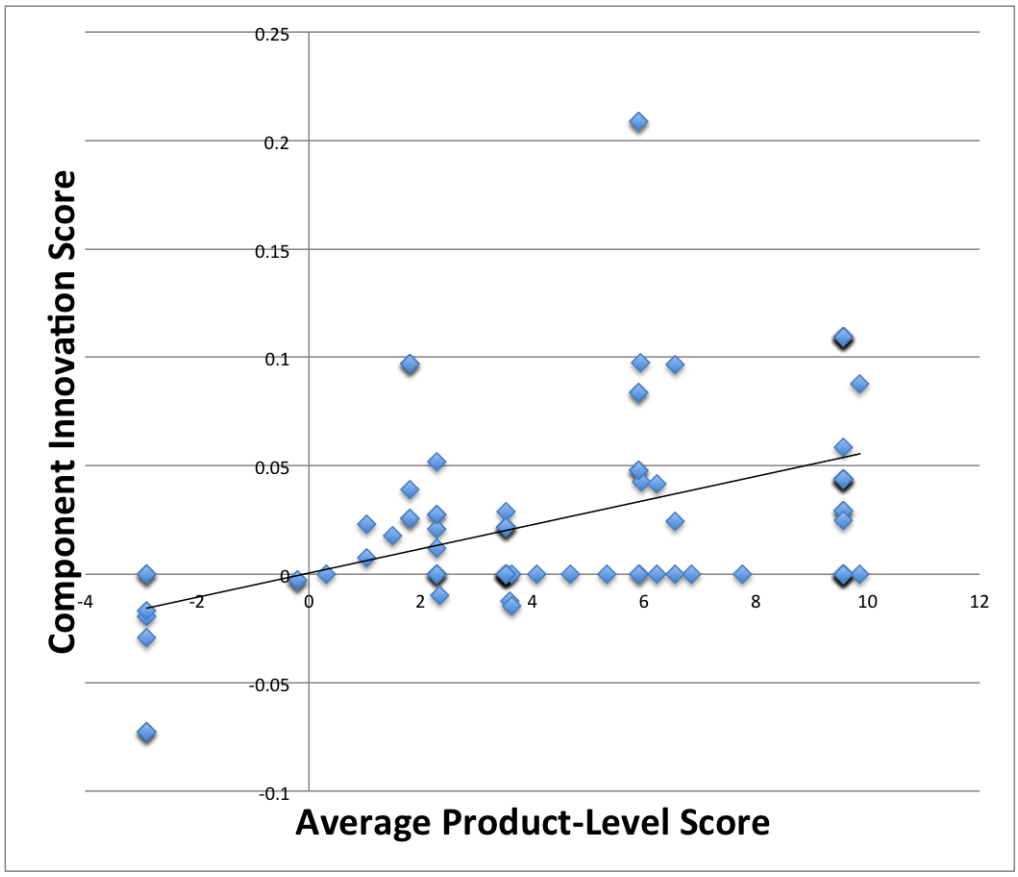

Figure 5.8: Agent-estimated values using the D^{least} reward on the latent variable dataset. There are two outliers in this data located at (5.9, 0.209) which represent the function-component pairs (channel energy, em sensor) and (channel energy,hydraulic pump)

Figure 5.8 shows that as the average product-level score increases, the innovation score assigned to the components tends to increase as well. These data are somewhat different from the data found in Figure 5.5 in that the innovation score peaks prematurely in relation to the product-level score. The tendency for the data to spread out as product-level score increases, but this trend is not pronounced.

The results shown in Figure 5.9 are collected generally around the zero point. This data still shows a trend which has been observed in the other results, which is that it

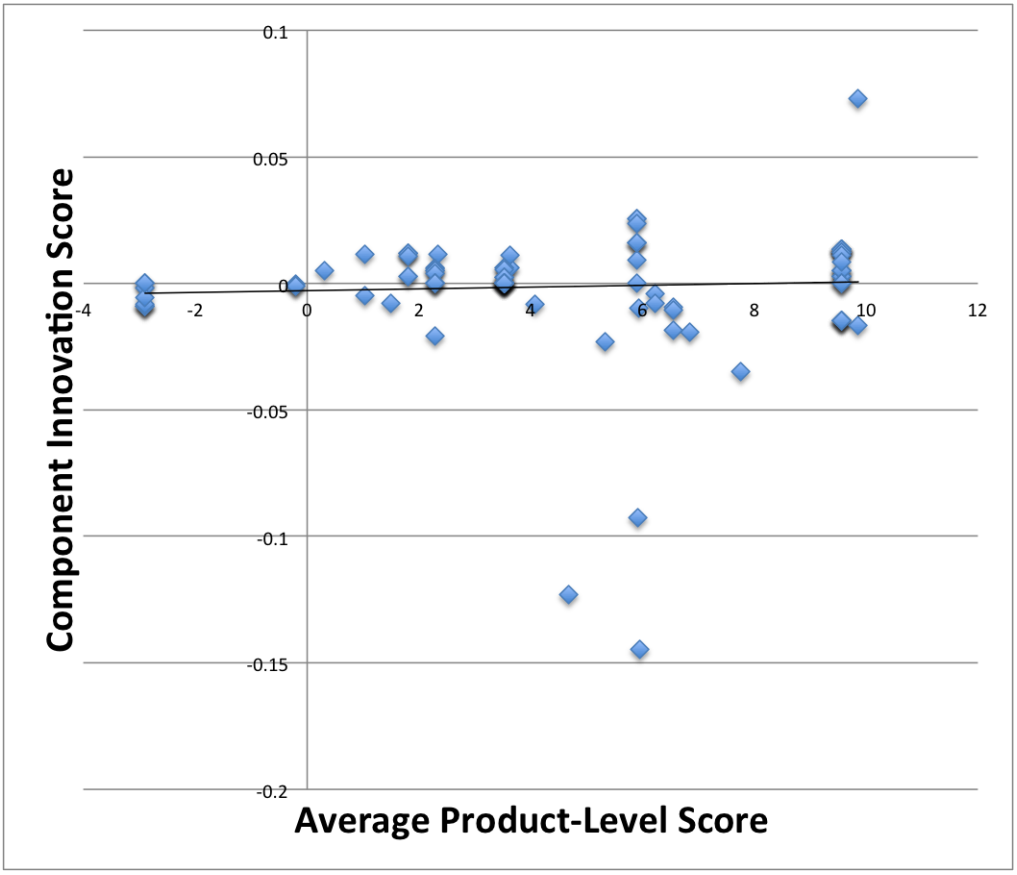

Figure 5.9: Agent-estimated values using the $D^{Average}$ reward on the latent variable dataset. This trends toward having a zero mean, but has several interesting outliers.

tends to spread out as the average product-level innovation increases. The data in Figure 5.9 also shows several outliers which are worth mentioning.

The point with the highest estimated innovation score is located on Figure 5.9 at (9.86,0.073) and corresponds to (signal signal, circuit board). What this is saying is not that products within this dataset tend to be more innovative if they *have* a circuit board, but that they tend to be more innovative if they *transmit a signal* using a circuit board.

So what do negative outliers mean? There are a group of three negative outliers obvious on Figure 5.9. Interestingly, two of them also involve a circuit board, but it is used in a different manner. The points at (5.94,-0.145) and (5.91,-0.093) correspond to

(channel energy, circuit board) and (channel signal, circuit board) respectively. The fact that these have a different effect not based on the component involved but also how it is used suggest that innovation may come from a creative application of a component rather than necessarily identifying a component as creative. Channeling energy and signals are an integral function of a circuit board, and therefore will appear wherever there is a circuit board in the design set. They perform a functional role but do not contribute to the innovation of a device. A circuit board which is transmitting a signal indicates that some kind of display was found on the product analyzed. A designer wishing to leverage this suggestion might attempt to incorporate a graphical display of some kind on a new product design.

Chapter 6: Analysis of the Results

Though there were differences in the source of the innovation scores as well as the method by which they were gathered, there are trends seen across all datasets which reflect some of our key findings in this work. In Section 6.1 we identify these trends and offer some observations about the data. However the main focus of this work is to develop a tool for application to design. Thus in Section **??** we demonstrate how to use this information to perform a redesign on a well-known product to promote higher innovation.

6.1 Trends in Innovation Scores for Components

Results across all datasets show a dip in novelty scores as well as a spreading out of the innovation scores as the product-level score increases. Novelty was generally too dispersed throughout the dataset to draw definite conclusions, but it appears that higher novelties in all cases tend to appear at the extreme values of average product-level score.

The innovation scores for function-component pairs obtained using D^{least} across all datasets showed an upward trend, generally learning that components in scores with a higher average product-level score had more innovation value than the those found in products with lower scores.

The scores found by $D^{average}$ may actually be more interesting. Using this reward, the data do not necessarily show an upward trend with the increase in average innovation score, but instead show the same dispersal of the data. This indicates that the difference reward is able to pull out what, in products with high scores, is both contributing and not contributing to those high scores.

Outliers in this data can give us some insight into components that are particularly influential in promoting innovation. Through an analysis of the outliers in the Latent Variable dataset using the $D^{average}$ scoring, we were able to identify that the presence of a component does not necessarily correlate with a better novelty score–how it is *used* contributes in large part to this score as well.

Looking at the outliers, the results may have been influenced in part by the audience

analyzing the dataset. We found that in the expert evaluation dataset we had two outliers which were (channel material,reservoir) and (control magnitude material,screw). These do not seem like particularly exciting components, but the innovation metrics in this case were derived from consumer magazines, which likely had a functionality and market influence. The function-component pairs found to be particularly innovative also are highly useful, and this usefulness may have played a part in their identification as 'innovative'.

The outlier identified in the survey data using $D^{average}$, (convert energy, handle) was also underwhelming. It was identified as having the highest novelty score, but this may say more about the population surveyed than its true innovation contribution. Again usefulness comes into play here; 'innovation' and 'usefulness' are hard to distinguish from one another in many cases, and the survey-takers were not design engineering students. When they were shown the designs, it was as a picture with a brief explanation, and not a full breakdown of how the product works. A handle is an easily-seen component which would show up in a picture, and would add to the usefulness of a design.

The latent variable data outliers, which *were* gathered from design engineering students, show more interesting function-component pairs identified as innovative. These were (channel energy, em sensor) and (channel energy, hydraulic pump). These identify components that have a higher technological complexity than the outliers of the other surveys, and therefore are potentially more interesting to design engineering students. It is likely that the design engineering students were able to better understand the functionality of a product, and therefore find it more innovative than someone who had only a functional or aesthetic knowledge of a product.

The influence of demographics on training data may also be a parameter that can be leveraged. Innovation *is* a subjective measure, and companies cannot design a product with all demographics in mind. Products need a market. If the demographics of this market can be used to bias the training data for the CARRIE algorithm, better evaluations of component innovation may be obtained for the target market.

6.2 Implication for Generating New Designs

Though we can demonstrate consistency and patterns in the data using our techniques, an example in an actual design application provides a more intuitive look at what the dif-

ferent techniques actually offer. For this reason, we present a redesign of the design which had the lowest innovation ranking according to our survey: the Dustbuster. We selected five functions in the Dustbuster with the lowest product-level average score ratings and calculated suggested replacements for the components according to the highest-ranked component solutions as discovered by our different techniques on the survey dataset. The results are shown in Table 6.2, which can be compared with the original design shown in Table 6.1.

The assessment of the innovation of the different replacements for the design is difficult to perform objectively. Additionally, as practicality is separated from the innovation, not all suggestions are necessarily optimal or even possible to implement. Nonetheless, all suggestions offer a designer a different perspective on how to modify a vacuum. The novelty evaluation suggests using a hydraulic pump, which is rarely if ever present in vacuums and may offer an interesting perspective on how to creatively channel material sucked into the vacuum. The D^{least} evaluation suggests that a speaker replace the electric switch to turn on the vacuum, which indicates a redesign featuring a voice-activated vacuum cleaner. The $D^{average}$ evaluation suggests that guiders may be used to replace the electric switch, suggesting that the vacuum might be designed to mechanically show when it is full. The material coming into the vacuum might brush past the guiders and put pressure on them which would activate another part of the design to detect fullness of the vacuum cleaner.

Ultimately the usefulness of the design suggestions is still in the hands of the engineer, but the suggestions based on D^{least} and $D^{average}$ appear to offer more interesting solutions overall to the given functions. They do not appear to be *practical*, necessarily, as many of the suggestions do not present feasible design decisions. This draws attention to the fact that 1) we are using the most abstract language in the design repository, which may mean that components designed to handle certain specific tasks may not be applicable for all functionalities and 2) we need some sort of method of identifying which components are *possible* to perform these functions. This makes MEMIC the ideal complement to this assessment technique, as its target is to identify component suggestions from designs which have had similar functional flows, and therefore this increases the chance that components may have similar functionality.

Function	Repository
convert energy to signal	electric switch
control magnitude energy	electric cord
channel energy	electric cord
channel material	guiders
convert energy	electric motor

Table 6.1: Five of lowest-scoring function-component solutions in the Dustbuster as estimated by the average product-level score. Items under 'Repository' are the original components performing the respective function in the Dustbuster.

Function	Novelty	D^{least}	$D^{average}$
convert energy to signal	light source	speaker	guiders
control magnitude energy	washer	abrasive	coupler
channel energy	abrasive	pulley	coupler
channel material	hydraulic pump	friction enhancer	shaft
convert energy	cover	magnitude controller	handle

Table 6.2: Redesign suggestions of five of lowest-scoring function-component solutions in the Dustbuster as estimated by the average product-level score. Items under 'Novelty', D^{least}, and $D^{average}$ are suggestions to perform the functions based on the novelty scores, D^{least} evaluation, and $D^{average}$ evaluation respectively. Ties between component scores are broken by the average product-level values.

Chapter 7: Conclusion

In this work we have developed the Creative Agents for Repository-Reliant Innovation Engineering (CARRIE) algorithm which uses difference rewards under a multiagent framework to propagate product-level innovation scores down to the component level. By framing the design process as a multiagent component-selection process with functional requirements modeled as agents, we are able to have our agents learn scores for component solutions after seeing several example designs. From the data gathered in this work, we can draw three conclusions; i) we identify trends in the novelty relating to our average product-level innovation score, validating our approach; ii) we can use this method to identify components that both add to and detract from the innovation score of the device; iii) we can use our evaluations to perform design alterations and give an indication of which components have historically increased innovation for a particular function.

Our first conclusion is validated by the tendency for the novelty scores to polarize, and typically decrease for mid-level average product-level scores. Novelty *does* relate to the innovation score of the device, but it is not directly proportional to innovation score. The trend toward a dip in novelty scores also uncovers a major shortcoming in our attempt to identify innovative components; the more common components may still receive a mid-level innovation score, but this depends on the *configuration* of components rather than the components themselves. This accounts for solutions in which common components are assembled in a creative configuration. We also demonstrate that in the high levels of average product-level innovation scores, more novel components are more frequently seen. We also see novel components in the lower levels of innovation–this is most likely due to devices which perform only one task uniquely. For example a microwave heats things in a way not found in any other product, but it only performs one task and therefore is not particularly innovative.

Our second conclusion best highlights the intent of developing our difference methods for innovation identification: in innovative products, there tend to be components which add more to the innovation of the device than other components. This comes with

the parallel observation that some components actually *detract* from the innovation of a product at higher product-level innovation. This invites the use of this method for design improvement toward higher levels of innovation, as we can identify which components help and hinder the innovation of a product and modify it accordingly.

The fact that we can identify components within innovative products that have a large positive or negative effect on the innovation of a design suggests two interesting findings: one, that the innovation of a product may be carried primarily by only a couple of components within the product, and two, that highly innovative products must rely on tried-and-true methods of performing functions in order to have a fully operational product. Because there are so many components which have negative scores under our D^{least} evaluation in the higher levels of average product-level score, this indicates that perhaps the most innovative products do *one* thing innovatively, and they do other things in a manner which is reliable. This is consistent with the previous research done using functional subtraction [17], which identified functions existing in innovative components which did not exist in the common components.

Innovation may not be desirable in every aspect of a design. If every function is solved in an unconventional manner, the resulting product may suffer because it is not *recognizable* as an improved version as an existing device, but instead as a different device. Additionally, there are many benefits to using known technology and standard subsystems, including the fact that the development time is shorter, the risks are known, and there already exists a market for a product with a basic set of known functionalities. Incorporating innovative suggestions into a design may give a product an edge in a known market, but it may be desirable to limit the focus of innovative solutions to a single set or subset of functionalities.

Our third conclusion was validated by our redesign of the Dustbuster as given in Section 6. By using the data that we had learned using the multiagent system, we were able to demonstrate the fact that the learned data provides an accessible way of obtaining design suggestions and their associated innovation scores. We also identified the fact that bias in our datasets plays a large role in what components will be considered the most innovative; both the products and the people evaluating them were different across the datasets, which introduced bias not only from the backgrounds of people, but by the different novelty scores for components in a different dataset. There may be a way to leverage this and target a more specific demographic and design subset, but this

is not explored in this work.

This work suggests that trends in component-level innovation can be found using a decomposition of the product-level innovation score, and offers motivation for more extensive testing and validation. Our novelty-based evaluation provides a framework for assessing the creative contribution of each component to the system score, but it is definitely not the only approach to the decomposition of this score. Given evaluations for a larger dataset than used in this work (such as all designs in the repository instead of just a subset of 50 or less), function approximation techniques may be employed to better break down the difference rewards without making the assumption that each component has an impact on the product-level score which is proportional to its system score.

One aspect of this work is that it does not consider the *performance* of the components in an application, just their innovativeness. When using the functional basis language with the most general terminology we may encounter situations where the component which is more innovative for a particular function will not work in the given task. There are two approaches that may be taken to solving this problem. The first is to obtain a larger dataset of product scorings, which would enable more specific language to be used, and increase potential of a component to be well-suited for a given (more specific) functionality. The second option is to combine this innovation scoring technique with a technique that balances innovation with performance. This may enter into the realm of multiobjective optimization and has not yet been explored, but currently techniques *do* exist for assessing whether a component will fit into a given structure of a device based on historical data. The current method of concept generation using this added consideration for how the design is structured is available in MEMIC. The scorings presented in this work can be used in conjunction with the MEMIC suggestion generator to better steer the designer toward making innovative component selections in the early stages of design.

We are the first to attempt this work, and there is no bar that we can reference for how close we were to the true innovativeness of the components. The *usefulness* of our measurements may be measured in the same way that the usefulness of concept generation techniques have been measured and validated. We have developed a tool which is intended to *inspire innovation* in engineers who use it. Though this technique seems theoretically sound from a multiagent perspective, the actual *usefulness* in introducing innovation into the design process must be decided by the engineers who use it.

Bibliography

[1] Biju Paul Abraham and Soumyo D Moitra. Innovation assessment through patent analysis. *Technovation*, 21(4):245 – 252, 2001.

[2] A. Agogino and K. Tumer. QUICR-learning for multiagent coordination. In *Proceedings of the 21st National Conference on Artificial Intelligence*, Boston, MA, July 2006.

[3] C. Bryant Arnold, R. Stone, and D. McAdams. MEMIC: An Interactive Morphological Matrix Tool for Automated Concept Generation. In *Proceedings of the 2008 Industrial Engineering Research Conference*, 2008.

[4] D. Borsboom, G. Mellenbergh, and J. van Heerden. The theoretical status of latent variables. *Psychological Review*, 110(2):203–219, 2003.

[5] Kees Dorst and Nigel Cross. Creativity in the design process: co-evolution of problemsolution. *Design Studies*, 22(5):425 – 437, 2001.

[6] Elizabeth C. Hirschman. Innovativeness, novelty seeking, and consumer creativity. *Journal of Consumer Research*, 7(3):pp. 283–295, 1980.

[7] D. Horn and G. Salvendy. Consumer-based assessment of product creativity: A review and reappraisal. *Human Factors and Ergonomics In Manufacturing*, 16(2):155–175, 2006.

[8] Karl K. Jeffries. Diagnosing the creativity of designers: individual feedback within mass higher education. *Design Studies*, 28(5):485 – 497, 2007.

[9] M. Knudson and K. Tumer. Robot coordination with ad-hoc team formation (extended abstract). In *Proceedings of the Ninth International Joint Conference on Autonomous Agents and Multiagent Systems*, Toronto, Canada, May 2010.

[10] J. Koza, M. Keane, M. Streeter, T. Adams, and L. Jones. Invention and creativity in automated design by means of genetic programming. *AI EDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18:245–269, 7 2004.

[11] Hong Liu and Mingxi Tang. Evolutionary design in a multi-agent design environment. *Applied Soft Computing*, 6(2):207 – 220, 2006.

[12] Y. Liu. Personal versus cultural cognitive models of design creativity. *International Journal of Technology and Design Education*, 8:185–195, 1998.

[13] D. MacFarland and L. Malta. Symptoms as latent variables. *Behavioral and Brain Sciences*, 33:165–166, 5 2010.

[14] M. Maher and J. Poon. Modelling design exploration as co-evolution. *The Special Issues of Microcomputers in Civil Engineering on Evolutionary Systems in Design*, 1996.

[15] Akira Miyake, Naomi P. Friedman, Michael J. Emerson, Alexander H. Witzki, Amy Howerter, and Tor D. Wager. The unity and diversity of executive functions and their contributions to complex frontal lobe tasks: A latent variable analysis. *Cognitive Psychology*, 41(1):49 – 100, 2000.

[16] B. O'Halloran, R. Stone, and I. Tumer. Link between Function-Flow Failure Rates and Failure Modes for Early Design Stage Reliability Analysis. In *In the Proceedings of IMECE2011-63110*, Denver, Colorado, USA, 2011.

[17] S. Oman, B. Gilchrist, C. Rebhuhn, I. Tumer, A. Nix, and R. Stone. Towards a Repository of Innovative Products to Enhance Engineering Creativity Education. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences and Computers and Information Engineering Conference IDETC/CIE 2012*, Chicago, Illinois, USA, August 2012.

[18] S. Proper and K. Tumer. Modeling difference rewards for multiagent learning (extended abstract). In *Proceedings of the Eleventh International Joint Conference on Autonomous Agents and Multiagent Systems*, Valencia, Spain, June 2012.

[19] J. Shah and N. Vargas-Hernandez. Metrics for measuring ideation effectiveness. *Design Studies*, 24(2):111–134, March 2003.

[20] R. Stone and K. Wood. Development of a functional basis for design. *Journal of Mechanical Design*, 122(4):359–370, 2000.

[21] K. Tumer and A. K. Agogino. Multiagent learning for black box system reward functions. *Advances in Complex Systems*, 12:475–492, 2009.

[22] K. Tumer and D. H. Wolpert. Collective intelligence and Braess' paradox. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 104–109, Austin, TX, 2000.

[23] Nash Unsworth and Gene A. Brewer. Individual differences in false recall: A latent variable analysis. *Journal of Memory and Language*, 62(1):19 – 34, 2010.

[24] YiLin Wong and KinWaiMichael Siu. A model of creative design process for fostering creativity of students in design education. *International Journal of Technology and Design Education*, 22:437–450, 2012.

[25] Shahid Yusuf. From creativity to innovation. *Technology in Society*, 31(1):1 – 8, 2009.

APPENDICES

Appendix A: Full Algorithm Flowchart

Figure A.1 shows a detailed flowchart of the CARRIE algorithm. Training data are taken from the repository and turned into functional requirements, component selection restrictions, and a product-level score. The functional requirements are then passed to the multiagent system, which activates a subset of these agents to participate in the design process. This subset is forced to select a restricted set of components. These components are collected together into (the original) design and are scored using the product-level score from the training data, which is either given as a binary score, a mean, or a latent variable. This product-level innovation score is then passed to the difference reward equation, which offers a set of rewards to each of the agents to reward their action selection. These agents then adjust their policies according to this reward, and the process is started again with another training design.

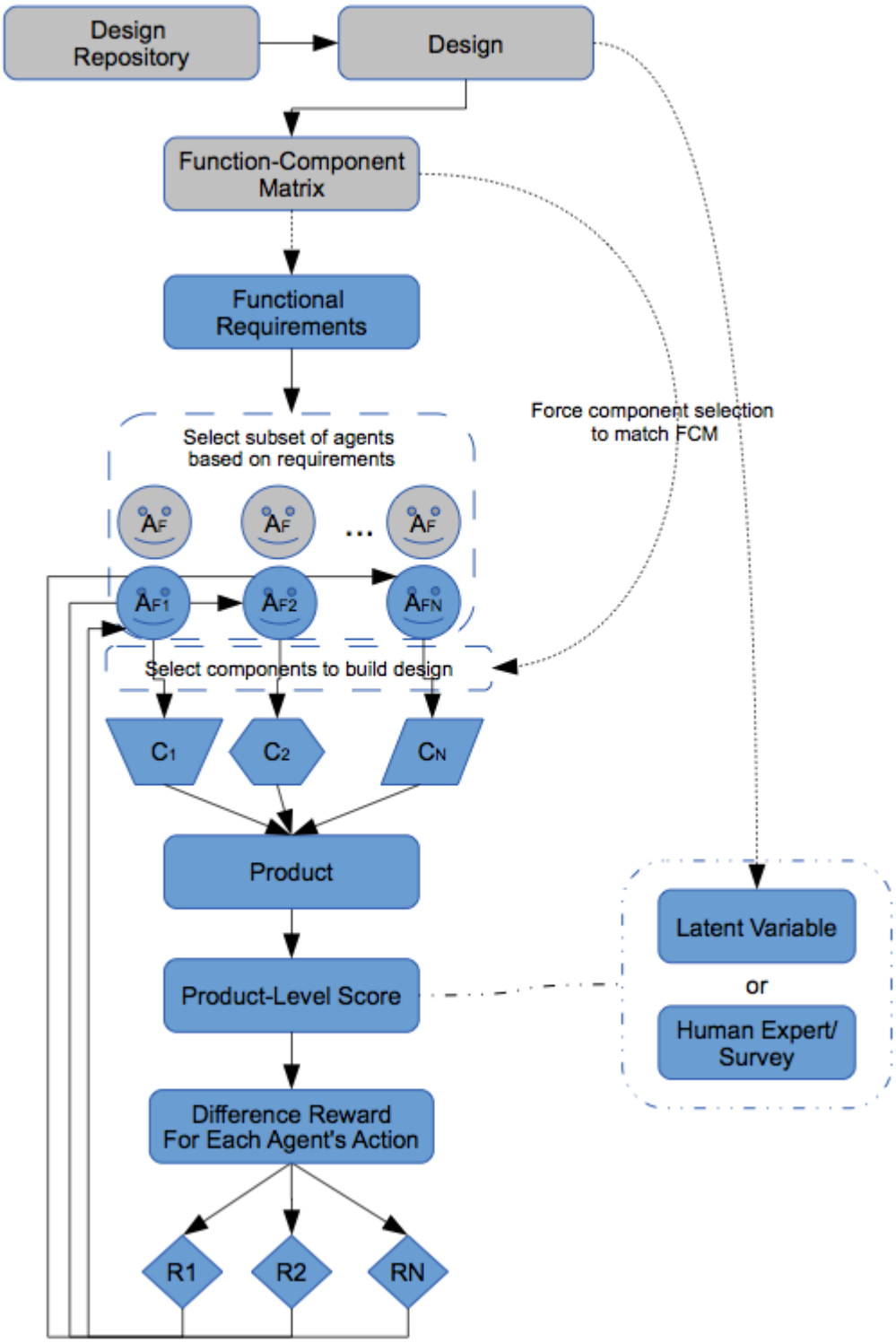

Figure A.1: The detailed overall flow of the CARRIE algorithm.

Appendix B: Expert Innovation Data

No.	Product	Identified as Innovative?
1	Random Orbital Sander	1
2	Black and Decker Palm Sander	0
3	Versapak Sander	0
4	DeWalt Sander	0
5	Delta Sander	0
6	Dyson Air Multiplier	1
7	Holmes Fan	0
8	Oliso Smart Iron	1
9	Proctor Silex Iron	0
10	E-sky Honeybee Helicopter	1
11	Air Hawg Toy Plane	1
12	Wowee Flytech Dragonfly	1
13	Bosch Brad Nailer	1
14	Hitachi Brad Nailer	0
15	Milwaukee Palm Nailer	1
16	Grip Rite Air Nailer	0
17	Clorox Ready Mop	1
18	Libman Wonder Mop	0
19	Microfiber Floor Mop	0
20	Kid Smart Smoke Detector	1

Table B.1: Table showing the expert product identification of innovation for products 1-20. The score under 'Identified as Innovative?' is binary (yes=1, no=0), and these numbers are used as a product-level score for the CARRIE algorithm.

No.	Product	Identified as Innovative?
21	First Alert Basic Smoke Alarm	0
22	Ridgid Jobmax	1
23	Craftsman Nextec Multi Tool	0
24	Dremel Multi Max	0
25	Milwaukee Copper Tubing Cutter	1
26	Neato Robotix Vacuum Cleaner	1
27	iRobot Roomba	0
28	Power Mat	1
29	Dual Powered Charging Station	0

Table B.2: Table showing the expert product identification of innovation for products 21-29. The score under 'Identified as Innovative?' is binary (yes=1, no=0), and these numbers are used as a product-level score for the CARRIE algorithm.

Appendix C: Informal Survey Data

No.	Product	Avg. Score	Std. Dev
1	Air Hawg Toy Plane	0.375	0.212459146
2	Alcohawk Digital Alcohol Detector	0.45	0.197202659
3	Apple USB Mouse	0.125	0.176776695
4	Black and Decker Drill Attachment	0.325	0.264837476
5	Black and Decker Dustbuster	0.025	0.079056942
6	Black and Decker Mini Router Attachment	0.5	0.166666667
7	Black and Decker Power Pack	0.275	0.248607232
8	Black and Decker Rice Cooker	0.075	0.120761473
9	Black and Decker Screwdriver	0.225	0.184466197
10	Black and Decker Slice Right	0.1	0.174801475
11	Ball Shooter	0.225	0.184466197
12	Bosch Brad Nailer	0.3	0.197202659
13	Braun Coffee Grinder	0.075	0.120761473
14	Bugvac	0.575	0.264837476
15	Burton Induction Cooktop	0.35	0.268741925
16	Fujifilm Single Use Camera	0.1	0.241522946
17	VHS Player	0.1	0.174801475
18	Sony Diskman	0.15	0.174801475
19	Chefn Palm Vegetable Peeler	0.6	0.174801475
20	Colgate Motion Toothbrush	0.25	0.263523138

Table C.1: Products 1-20 of the set used in the survey data. The Avg. Score value is used as a product-level evaluation of innovation in the CARRIE algorithm.

No.	Product	Avg. Score	Std. Dev
21	Coolit Drink Cooler	0.65	0.241522946
22	Cordless Kettle	0.275	0.184466197
23	Rose Art Cotton Candy Machine	0.4	0.210818511
24	Craftsman Nextec Multi Tool	0.475	0.218898759
25	Dazey Stripper	0.575	0.264837476
26	Delta Jigsaw	0.125	0.131761569
27	Delta Nail Gun	0.15	0.174801475
28	DeWalt Sander	0.1	0.174801475
29	Digger Dog	0.75	0.333333333
30	Digital Scale	0.05	0.105409255
31	Dirt Devil Vacuum	0.075	0.120761473
32	Dishwasher	0.65	0.293446948
33	Dremel Multi Max	0.35	0.210818511
34	Dryer	0.075	0.168737139
35	Dual Powered Charging Station	0.475	0.248607232
36	Durabrand Iron	0.025	0.079056942
37	Sony DVD Player	0.15	0.174801475
38	Dyson Air Multiplier	0.8	0.197202659
39	E-sky Honeybee Helicopter	0.325	0.120761473
40	Electric Stapler	0.325	0.205818151
41	Automatic Eyeglass Cleaner	0.55	0.197202659
42	Black and Decker Firestorm Flashlight	0.175	0.168737139
43	Black and Decker Firestorm Saber Saw	0.25	0.288675135
44	First Shot Nerf Gun	0.225	0.184466197
45	Shake n Go Racer Dog	0.3	0.25819889
46	Game Controller	0.2	0.197202659
47	GE Microwave	0.175	0.237170825
48	GSE Solar Power Module 60W	0.625	0.412478956
49	Hair Trimmer	0.3	0.158113883
50	Popcorn Popper	0.2	0.197202659

Table C.2: Products 21-50 of the set used in the survey data. The Avg. Score value is used as a product-level evaluation of innovation in the CARRIE algorithm.

Appendix D: Latent Variable Data

This represents a segment of the data that a collaborator collected for 8 products.

No.	Product	Innovation Latent Variable Score
1	Dyson	10.16
2	Holmes Fan	-2.90
3	Power Mat	5.90
4	Dual Powered Charging Station	1.80
5	Oliso Smart Iron	9.56
6	Proctor Silex Iron	3.53
7	KidSmart Smoke Detector	2.29
8	First Alert Basic Smoke Alarm	-0.21

Table D.1: Products with latent variable scoring. The Avg. Score value is used as a product-level evaluation of innovation in the CARRIE algorithm.

Bibliography

[1] Biju Paul Abraham and Soumyo D Moitra. Innovation assessment through patent analysis. *Technovation*, 21(4):245 – 252, 2001.

[2] A. Agogino and K. Tumer. QUICR-learning for multiagent coordination. In *Proceedings of the 21st National Conference on Artificial Intelligence*, Boston, MA, July 2006.

[3] C. Bryant Arnold, R. Stone, and D. McAdams. MEMIC: An Interactive Morphological Matrix Tool for Automated Concept Generation. In *Proceedings of the 2008 Industrial Engineering Research Conference*, 2008.

[4] D. Borsboom, G. Mellenbergh, and J. van Heerden. The theoretical status of latent variables. *Psychological Review*, 110(2):203–219, 2003.

[5] Kees Dorst and Nigel Cross. Creativity in the design process: co-evolution of problemsolution. *Design Studies*, 22(5):425 – 437, 2001.

[6] Elizabeth C. Hirschman. Innovativeness, novelty seeking, and consumer creativity. *Journal of Consumer Research*, 7(3):pp. 283–295, 1980.

[7] D. Horn and G. Salvendy. Consumer-based assessment of product creativity: A review and reappraisal. *Human Factors and Ergonomics In Manufacturing*, 16(2):155–175, 2006.

[8] Karl K. Jeffries. Diagnosing the creativity of designers: individual feedback within mass higher education. *Design Studies*, 28(5):485 – 497, 2007.

[9] M. Knudson and K. Tumer. Robot coordination with ad-hoc team formation (extended abstract). In *Proceedings of the Ninth International Joint Conference on Autonomous Agents and Multiagent Systems*, Toronto, Canada, May 2010.

[10] J. Koza, M. Keane, M. Streeter, T. Adams, and L. Jones. Invention and creativity in automated design by means of genetic programming. *AI EDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18:245–269, 7 2004.

[11] Hong Liu and Mingxi Tang. Evolutionary design in a multi-agent design environment. *Applied Soft Computing*, 6(2):207 – 220, 2006.

[12] Y. Liu. Personal versus cultural cognitive models of design creativity. *International Journal of Technology and Design Education*, 8:185–195, 1998.

[13] D. MacFarland and L. Malta. Symptoms as latent variables. *Behavioral and Brain Sciences*, 33:165–166, 5 2010.

[14] M. Maher and J. Poon. Modelling design exploration as co-evolution. *The Special Issues of Microcomputers in Civil Engineering on Evolutionary Systems in Design*, 1996.

[15] Akira Miyake, Naomi P. Friedman, Michael J. Emerson, Alexander H. Witzki, Amy Howerter, and Tor D. Wager. The unity and diversity of executive functions and their contributions to complex frontal lobe tasks: A latent variable analysis. *Cognitive Psychology*, 41(1):49 – 100, 2000.

[16] B. O'Halloran, R. Stone, and I. Tumer. Link between Function-Flow Failure Rates and Failure Modes for Early Design Stage Reliability Analysis. In *In the Proceedings of IMECE2011-63110*, Denver, Colorado, USA, 2011.

[17] S. Oman, B. Gilchrist, C. Rebhuhn, I. Tumer, A. Nix, and R. Stone. Towards a Repository of Innovative Products to Enhance Engineering Creativity Education. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences and Computers and Information Engineering Conference IDETC/CIE 2012*, Chicago, Illinois, USA, August 2012.

[18] S. Proper and K. Tumer. Modeling difference rewards for multiagent learning (extended abstract). In *Proceedings of the Eleventh International Joint Conference on Autonomous Agents and Multiagent Systems*, Valencia, Spain, June 2012.

[19] J. Shah and N. Vargas-Hernandez. Metrics for measuring ideation effectiveness. *Design Studies*, 24(2):111–134, March 2003.

[20] R. Stone and K. Wood. Development of a functional basis for design. *Journal of Mechanical Design*, 122(4):359–370, 2000.

[21] K. Tumer and A. K. Agogino. Multiagent learning for black box system reward functions. *Advances in Complex Systems*, 12:475–492, 2009.

[22] K. Tumer and D. H. Wolpert. Collective intelligence and Braess' paradox. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 104–109, Austin, TX, 2000.

[23] Nash Unsworth and Gene A. Brewer. Individual differences in false recall: A latent variable analysis. *Journal of Memory and Language*, 62(1):19 – 34, 2010.

[24] YiLin Wong and KinWaiMichael Siu. A model of creative design process for fostering creativity of students in design education. *International Journal of Technology and Design Education*, 22:437–450, 2012.

[25] Shahid Yusuf. From creativity to innovation. *Technology in Society*, 31(1):1 – 8, 2009.